Representation, Learning, Generalization and Damage in Neural Network Models of Reading Aloud

John A. Bullinaria

Neural Networks Research Group
Department of Psychology
University of Edinburgh
7 George Square
Edinburgh EH8 9JZ
Scotland.

j.bullinaria@physics.org

Abstract

We present a new class of neural network models of reading aloud based on Sejnowski & Rosenberg's NETtalk. Unlike previous models, they are not restricted to mono-syllabic words, require no complicated input-output representations such as Wickelfeatures and require no pre-processing to align the letters and phonemes in the training data. The best cases are able to achieve perfect performance on the Seidenberg & McClelland training corpus (which includes many irregular words) and in excess of 95% on a standard set of pronounceable non-words. Evidence is presented that relate the output activation error scores in the model to naming latencies in humans. Several possible accounts of developmental surface dyslexia are identified and on various forms of damage the models exhibit symptoms similar to acquired surface dyslexia. However, their inability to account for lexical decision, the pseudohomophone effect and phonological dyslexia indicate that we will still need to introduce an additional lexical/semantic route before we have a complete model of reading aloud. Nevertheless, the models' simplicity, performance and room for improvement make them a promising basis for the grapheme-phoneme conversion route of a realistic dual route model of reading.

Edinburgh University Technical Report 94/1 – November 1994

Introduction

There are many processes involved in the act of 'reading aloud' and we are clearly a long way from being able to construct a realistic model of them all. Any complete reading system must (at least) be able to :

1. recognise whole words and groups of words, parse them, relate them to their meanings and output their sounds, and

2. read aloud unknown words or pronounceable non-words (i.e. have a series of rules that can simply convert text to phonemes),

and there is currently a lively debate concerning the exact mental processes underlying these abilities. The traditional position (recently championed by Coltheart, Curtis, Atkins & Haller, 1993) is that reading can only be described by a dual route model, with one route consisting of a series of grapheme to phoneme rules (necessary for reading new words or pronounceable non-words) and another route that involves some form of lexicon (necessary to output phonemes for irregular/ exception words which do not follow the rules and to provide a contact point for semantics and traditional natural language processing). An alternative view is that a single route may be sufficient and explicit neural network models of reading have been constructed (e.g. Plaut, McClelland & Seidenberg, 1992) that are able to learn all the words (including irregular words) in their training data and also read new non-words with accuracies comparable to human subjects. Although it seems unlikely, at this stage, that a single route model will be able to account for all aspects of human reading abilities (Coltheart et al., 1993), there is considerable evidence that the two routes of the traditional dual route model can not be totally independent (Humphreys & Evett, 1985).

There are several directions from which we can attack the problem of understanding reading. The approach we propose here is to consider in some detail the possibilities for constructing explicit single route connectionist models of text to phoneme conversion and then examine how well these can fit in with more complete models of reading.

Coltheart et al. (1993), as part of their dual route model, have already developed an explicit (non-neural network) rule based text to phoneme system that has good generalization performance, but (by construction) is poor at reading the irregular words in the original training set. The class of neural network models presented in this paper involve no explicit lexicon and no representation of semantics and might therefore be considered to be an explicit neural network implementation of this Grapheme Phoneme Conversion (GPC) route of a dual route model. However, neural network models are generally able to map between word segments larger than single graphemes and phonemes. Thus, given that an exception word mapping can be thought of as a very low frequency high powered rule (i.e. a rule that is activated only for one specific word and over-rides all other potentially useful rules) such models should be able to handle exceptional words as well. Regular words will be pronounced according to simple rules, exception words will be pronounced according to complicated special purpose rules (effectively a lexicon) that must over-rule the simpler rules. There will clearly be a continuous spectrum between these two classes of words and since there are very few (if any) 'exception' words that do not contain any regular features at all, the need for true

lexical entries will be minimal. The success of the model depends on the network maximizing its use of simple rules whilst minimizing its use of special purpose rules. In this way, when presented with new words or non-words, none of the special purpose rules will fire and the network will output phonemes according to a full set of regular (GPC) rules, yet it will still be able to pronounce the exceptional words it has been trained on.

The rest of this paper will explore the possibilities for constructing such connectionist models of reading and present detailed performance results for one particularly promising class of models. We begin with a discussion of the representational problems underlying all connectionist models of reading and propose that the best way to proceed is to build upon the early and rather successful NETtalk model of Sejnowski & Rosenberg (1987). We then consider in some detail what changes need to be made to the original NETtalk model. In particular, we show how the learning algorithm can be modified to obviate the need to pre-process the training data and how simple context flags can be used to deal with the problem of homograph ambiguity. A range of variations on this basic approach are then investigated and we eventually identify the most promising model of this type. The performance of this model (in particular, the reading development, non-word reading and naming latencies) is then compared with that of humans. We then examine the extent to which we can understand developmental and acquired dyslexias within these models. Finally, we discuss the fundamental limitations of such single route connectionist models, and suggest the way forward for the connectionist modelling of reading.

Representation

The first thing that has to be decided for any model of reading is the representation to use for the inputs (letters) and outputs (phonemes). This section discusses this choice in some detail. The simplest way to proceed with a neural network would be :

1. Have *inchar* sets of input units (where *inchar* is the number of characters, i.e. letters, in the longest word to be presented) and have *nletters* units in each set (one for each different letter in the alphabet, e.g. 26 for English). Then, for each word, have the first letter activate the appropriate unit in the first set, the second letter activate the appropriate unit in the second set, and so on.

2. Have *outchar* sets of output units (where *outchar* is the number of characters, i.e. phonemes, corresponding to the longest word to be presented) and have *nphoneme* units in each set (one for each different phoneme used in the training data, e.g. about 40 in English). Then, for each word, have the first phoneme activate the appropriate unit in the first set, the second phoneme activate the appropriate unit in the second set, and so on.

3. Have a set of connections and hidden units between the inputs and outputs and use some training algorithm to adjust the connection weights so that the activation of each word on the input units produces the correct pronunciation on the output units.

There are a number of problems with this scheme. First, the network will have to

learn a separate set of letter to phoneme rules for each of the *inchar* sets of input units and this is not very efficient. This would not be so bad if there were a one-to-one correspondence between the letters and phonemes of every word; it would then be fairly easy to set up a neural network to map from letter strings to phoneme strings, and we would end up with a set of *inchar* versions of the required GPC rules. Unfortunately, however, the letter-to-phoneme correspondence is often many-to-one (up to four letters can map to one phoneme in English, e.g. 'ough' → /O/ in 'though') and sometimes one-to-many (e.g. 'x' → /ks/ in 'box'). (We use the phonemic notation and terminology of Seidenberg & McClelland (1989) throughout.) This means that the phoneme position corresponding to a given letter position will be different for different words. This is not only disastrous for the efficiency but also for the generalization ability. What has been learnt about a particular letter in one position will not generalize to the same letter in another position, nor even the same letter in the same position in a different word. Clearly, more sophisticated models are necessary.

One of the first successful neural network systems to get round this problem was NETtalk by Sejnowski & Rosenberg (1987). They made two important changes to the above procedure. First, to prevent duplication, instead of having the network process whole words all at once, they effectively processed them one letter at a time, reducing *outchar* to one. A window of *inchar* characters moves across the input words and activates the appropriate input units and the network is trained to output the phoneme corresponding to the letter in the centre of that window. This meant that only one set of GPC rules was being learnt and consequently we could expect good generalization. Secondly, to align the letters and phonemes, they pre-processed the training data by inserting special continuation (i.e. no output) characters into the phoneme strings. The introduction of the moving window was an important step forward and the model worked very well, but for many, the need to pre-process the training data by hand was considered unacceptable since it involved the system designer doing a significant amount of the work the network should be doing for itself.

This led Seidenberg & McClelland (1989) to use a radically different representation involving a system of distributed Wickelfeatures in which each letter and phoneme string is split into sets of triples of characters (Wickelgren, 1969) in the same way as for earlier models of past tense acquisition (Rumelhart & McClelland, 1986). This certainly removed the dependence on the absolute positions of the input letters and also bypassed the problem of aligning the letters and phonemes in the training data, but it made the interpretation of the networks output somewhat complicated and presented enormous difficulties in understanding the nature of the internal representations. This model was also restricted to mono-syllabic words and had an unacceptably poor generalization performance (Besner, Twilley, McCann & Seergobin, 1990).

More recent neural network models by Plaut, McClelland & Seidenberg (1992) and Plaut & McClelland (1993) use yet another representation. They showed that 108 orthographic input units (one for each of the Venezky graphemes occurring in the initial consonant, vowel and final consonant clusters) and 57 phonological output units are sufficient to represent virtually all uninflected monosyllables. Having only one set of input and output units for each of the three clusters brings us a long way towards the efficiency of the NETtalk representation and hence these models do very well at learning the training data and in generalizing to new words

or non-words. However, they are still restricted to mono-syllabic words and one might still argue that too much work is being done by the person choosing the representation and not enough by the network itself. A related multiple-levels interactive activation network has been proposed by Norris (1993) and attracts similar criticism.

In this paper we will investigate the possibility that these new and relatively complicated representations of Seidenberg, McClelland and Plaut are not necessary and that straightforward extensions of the original and conceptually simpler approach of Sejnowski and Rosenberg can solve all our problems. The NETtalk model has several unattractive features: We shall discuss the finite sized moving window later. First we re-consider the need to pre-process the training data.

Given only a single training example, e.g. 'ace' → /As/, there are many possible letter to phoneme rules we could use, namely:

a	→	As	A	A	_	_	_
c	→	_	s	_	As	A	_
e	→	_	_	s	_	s	As

and these six ways of representing the output are all equally valid. However, given a whole set of training examples we want a minimal set of letter to phoneme rules that are applicable as generally as possible. To someone with a good knowledge of English pronunciation, it is clear that the rules 'a' → /A/, 'c' → /s/ and 'e' → /–/ are going to be more generally applicable than 'a' → /–/, 'c' → /A/ and 'e' → /s/ or any of the other possibilities. NETtalk has to be given the best alignment in its pre-processed training data. If, for each word, our neural network was also able to choose (for itself) from all the possibilities which was the best way to represent the output, then our representational problems would be solved. Fortunately, it has recently been demonstrated (Bullinaria, 1993b) that under certain circumstances (including those applicable here) a neural network *is* able to choose its own representation from a set of possibilities. All we need to do then is combine this procedure with the Sejnowski & Rosenberg (1987) system and we should end up with a successful model of reading aloud.

The Basic Model

The preceding discussion thus leads us to our basic NETtalk style model which will be described in this section. We will see, however, that in addition to modifying the NETtalk learning algorithm, there are actually several other changes that we need to make.

The Architecture and Learning Algorithm

The basic architecture consists of a standard fully connected feedforward network (as shown in Figure 1) with sigmoidal activation functions and one hidden layer set up in the same way as the NETtalk model of Sejnowski & Rosenberg (1987). The input layer consists of a window of *nchar* sets of units, each set consisting of one unit for each letter occurring in the training data (i.e. 26 for English). The output layer consists of one unit for each phoneme occurring in the training data (i.e. about 40

units). The input words slide through the input window which is *nchar* letters wide, starting with the first letter of the word at the central position of the window and ending with the final letter of the word at the central position. Each letter activates a single input unit.

The original Sejnowski & Rosenberg (1987) model relied on there being a one-to-one correspondence between the letters and the phonemes for each word. The activated output phoneme for each presentation of the word then corresponded to the letter occurring in the centre of the window in the context of the other letters occurring within the window. To guarantee that each phoneme string was no longer than the corresponding letter string, a phonemic convention was used whereby no letter ever gave rise to more than one phoneme. We have already noted above that this is not true for other conventions, in particular that used in the Seidenberg & McClelland (1989) data set which has become the standard for training models of reading. Here three of the letters (namely 'j', 'g' and 'x') can correspond to more than one phoneme (namely /dZ/, /dZ/ and /ks/) and hence result in some words having more phonemes than letters (e.g. 'box' → /boks/). In order to solve this problem without abandoning the Seidenberg & McClelland corpus and yet keep our basic model as simple as possible, the combinations /dZ/ and /ks/ were simply replaced by additional 'phonemes' /J/ and /X/ respectively, bringing the total number of phonemes up to forty. We shall see later how a variation of our basic model can solve this problem without the need for such recoding. Since there can also be a many-to-one correspondence between the letters and phonemes, some of the outputs must be blanks (i.e. no phoneme output). It was the problem of having to insert the blanks into the training data by hand that hampered progress with this type of model in the past.

The simple alternative to inserting them by hand, that we are proposing here, is effectively to generate training data such that the set of phonemes corresponding to each word is padded out with blanks (to the same number of phonemes as there are letters in the word) in all possible ways. If there are *nl* letters and *np* phonemes, then there are $ntarg = nl! / np! (nl - np)!$ ways that this can be done. Clearly, we only want to train the network on one of these *ntarg* possible targets or we will run into the well known problems of over-fitting noisy training data. The important discovery is that by calculating for each input word the total error corresponding to each of the possible targets and only propagating back the error from the target with the least error, the network is (with a suitably diverse set of training words) able to *learn* which is the appropriate target for each word. This multi-target approach to neural network training is discussed in more detail in Bullinaria (1993a,b).

For example, consider the word 'ace' again and the corresponding phonemes /As/. This training example will be presented $nl = 3$ times, each with $ntarg = 3$ possible target outputs:

presentation	inputs	target outputs
1.	- - - **a** c e -	A A -
2.	- - a **c** e - -	s - A
3.	- a c **e** - - -	- s s

For each of the three input presentations the network's output activation error is calculated for each of the three target outputs. The sum of the errors for each target over the three input presentations is then computed and the target with the

minimum total error is used to update the network weights in the appropriate manner. With a realistic set of training patterns the regular correspondences such as 'a' → /A/ and 'c' → /s/ will dominate the weight changes over others such as 'c' → /A/ and 'e' → /s/ and eventually the network learns that the appropriate target is /As–/ rather than /A–s/ or /–As/.

Simulation Details

The networks were trained using the standard back-propagation gradient descent algorithm (Rumelhart, Hinton & Williams, 1986) with the extended Seidenberg & McClelland training corpus of 2998 monosyllabic words consisting of the original Seidenberg & McClelland (1989) set based on that of Kucera & Francis (1967) plus 101 other words missing from that set (Plaut et al., 1992). The small initial weights were chosen randomly with a rectangular distribution in the range –0.1 to +0.1. The learning algorithm parameters suffered two conflicting constraints. If they were set too low, the network training time became prohibitive. If they were set too high, the network failed to learn properly. After some experimentation, the back-propagation learning rate was therefore fixed at 0.05 and the momentum factor at 0.9 (which also happen to be the same values as used by Seidenberg & McClelland, 1989). Over-learning was controlled by not propagating back the error signal for words that already had the correct phoneme outputs and a total output activation *err*or less than some *crit*ical threshold *errcrit.* This also had the important side effect of reducing the training time.

Another important constraint placed upon us by the need to keep the training time down, was in our use of word frequencies in the training data. The frequency distribution of the words used during training has to be in line with that experienced when humans are learning because there are several important frequency effects we are hoping to model. However, the Kucera & Francis (1967) word frequencies for our training data range from 1 to about 67000, meaning that if we presented the words in each epoch with probabilities proportional to their actual frequencies, we would have to train our networks for hundreds of thousands of epochs before all the low frequency exception words could be learnt. This was clearly totally infeasible. Fortunately, Seidenberg & McClelland (1989) found that logarithmically compressing the frequency distribution in their training data did not appear to significantly reduce the frequency effects in their results. Consequently, we dealt with the training data in the same way for our model. The data was presented in each epoch in random order with probability given by the same logarithmic frequency distribution as in Seidenberg & McClelland (1989), resulting in a reduced probability range of about 0.058 to 0.930 and about 25% of the training data being used in each epoch.

Once trained there are several ways we can interpret the network's outputs. We shall use the most straightforward, in which the output phoneme of the network is simply defined to be the phoneme corresponding to the output unit with the highest activation. More sophisticated versions in the future will undoubtedly benefit from the imposition of more complicated decision criteria, basins of attraction, etc. (e.g. as in Hinton & Shallice, 1991; Plaut & McClelland, 1993).

During each training simulation the network's performance on a number of data sets was recorded. The most obvious of which were the percentages learnt correctly and the mean square errors for the full training set and the validation data

set. For more detailed analysis of the networks performance, similar data was also recorded for various interesting subsets of the training data (e.g. high and low frequency regular words, exception words, inconsistent words, strange words, homographs, etc.). For ease of comparison with studies of humans and other models, we utilized the precise subsets and controls defined and used by Taraban & McClelland (1987) and also used by Seidenberg & McClelland (1989).

Due to the large amount of computer processing power required for these simulations, only about 30 fairly small runs of this basic model have so far been carried out and it is often difficult to distinguish real improvements caused by parameter changes from statistical fluctuations. Some preliminary results have already been presented in Bullinaria (1993a).

Validation Data : Non-Words

To test the networks' generalization ability (i.e. their success at learning the GPC rules) we measured the extent to which the networks produced 'acceptable' pronunciations for three sets of non-words. As in Plaut et al. (1992), the three sets of non-words used were the 43 regular non-words and 43 exception non-words of Glushko (1979, Experiment 1) and the 80 control non-words of McCann & Besner (1987, Experiment 1). The allowable pronunciations of these non-words were derived from the training data by matching word segments (particularly rimes) in the non-words with the same segments in the training data and constructing possible non-word pronunciations by concatenating the pronunciations of the segments from the training data. For the regular non-words this typically led to a single allowable pronunciation (e.g. 'doon' →/dUn/ by analogy with 'soon' → /sUn/), but for the exceptional non-words there were often several allowable pronunciations (e.g. 'hove' → /hOv/ as in 'cove', → /h^v/ as in 'love' , → /hUv/ as in 'move'). For comparison with other studies we also distinguished between the strictly regular pronunciations (such as 'hove' → /hOv/) and the other acceptable pronunciations.

Problems Learning the Exceptions

The first few simulations were somewhat disappointing since the networks' performance levelled off at about 85% of the training data. A closer investigation of the networks outputs revealed that the 85% overall corresponded to 93% of the regular words but only 45% of the irregular words and that this situation had arisen because the network had started off by acquiring the main rules quite successfully but was then unable to replace these where appropriate by the less common sub-rules and exceptions. This is actually a quite common problem with the back-propagation learning algorithm because the error signal propagated back through the network is proportional to the derivative of the sigmoid which tends to zero as the output activations go totally wrong. Consequently, once an output has been learnt incorrectly, the network has difficulty in correcting it. In order to prevent their activations getting stuck hard wrong like this, Seidenberg & McClelland (1989) used output targets of 0.1 and 0.9 instead of 0.0 and 1.0. Another common solution is to use a Sigmoid Prime Offset of 0.1 instead (Fahlman, 1988). Figures 2 and 3 compare these two approaches for the case of 40 hidden units and a window size of

7 characters. Both approaches show a significant improvement over the unmodified learning algorithm. The final generalization performance was similar for the two approaches, but the Sigmoid Prime Offset (SPO) resulted in a superior learning rate and was therefore used throughout the remainder of the simulations. The original poor performance on the exception words but near normal performance on the regular words and non-words is very reminiscent of certain types of developmental dyslexia and will be discussed further in the section devoted to learning and developmental dyslexia.

Dealing with the Homographs

Since the Seidenberg & McClelland corpus contains 13 pairs of homographs it is clear that the network can never achieve total success at learning this training data and it is well known that such ambiguities can cause serious problems with neural network learning and generalization (e.g. Bullinaria, 1993b). In humans we have no trouble making use of (semantic) context information to resolve these ambiguities. Experiments were therefore carried out on the use of context flags to resolve these ambiguities for our networks. As a preliminary investigation, this was implemented in the simplest manner possible by introducing extra characters into the input alphabet and appending those characters to the least regular input word of each pair of homographs. The aim was to see how many different input flags would have to be introduced to deal with all 13 homograph ambiguities. It turned out that a single extra character was sufficient and that this not only allowed the network to achieve 100% success rate on the homographs (compared with a maximum of 50% before) but also resulted in improved performance on many orthographically similar non-homographs as well. That such a simple flag works so well gives us hope that similar flags could also be used to flip the network between different accents and languages as effortlessly as in humans.

Clearly, this simplistic approach to dealing with homographs will have to be re-assessed once we have a better understanding of how the model performs and how we might incorporate semantic processing into it. We will therefore discuss this issue further in our concluding section.

Results from the Simulations

For the basic model described above we will merely outline the gross features of its performance. A detailed analysis of what structures are actually being represented in the hidden units, the relationship between the output activation errors and naming latencies, developmental dyslexias and how the model responds to different types of damage will be reserved for the more successful and realistic variations of this model to be discussed in the next section.

There are still a few more network parameters that we can vary. The first thing we needed to consider was the appropriate window size and number of hidden units. As with the original NETtalk model, various values for these parameters were tried. We began with 40 hidden units (since this was the number of output units) and increased the widow size from one character (which not surprisingly did not work too well) until we got acceptable performance. Networks with a window size of 9 characters were able to learn all but one of the training examples, namely

'though' → /DO/. The reason why the network failed on this word was that the training data also included the word 'thought' → /T*t/ in which the sub-word 'though' had to be pronounced as /T*/ and unless the input window is large enough to have the final 't' in the window while the initial 't' is in the centre of the window, the network has no way of resolving the ambiguity. By increasing the window size to 13 this long range dependency could be handled and the network was able to achieve a 100% success rate on its training data.

To confirm the networks' capability of handling long range dependencies and also to test its ability to deal with more complex multi-syllabic words, some runs with the words 'photographic' → /fotOgrafik/ and 'photography' → /fot*grafE/ incorporated into the training data were carried out. Each of these words contain the letter 'o' pronounced in two different ways and the pronunciation of the second 'o' is determined only by letters at least six characters away. With a window size of 13 characters the network was able to learn both words without any difficulty. With a window size of only 11 characters (for which the crucial 'i' and 'y' fall outside the window while the problematic second 'o' is in the central position) the network failed to learn the two words.

Clearly there is nothing to stop us inventing new words with arbitrarily long range dependencies, so the required window size is very much training data dependent. This is obviously a fundamental limitation of this model which will be discussed further in our concluding section. However, having determined that, for our training data, we require a window size of 13 characters we could set about finding the optimal number of hidden units. A whole range of numbers of hidden units were tried and Figure 4 summarizes how the training performance varied. The general trend was that the more hidden units the network had, the less epochs it required to learn the training data. This reduction in number of epochs is, of course, counter-balanced (on serial computers) by the increased time taken to compute each epoch so this in itself is not an important criterion. In fact, in terms of total training time, the less hidden units the better. For less than about 30 hidden units the network fails to learn all the training data so we need to keep well above that. For our purposes it is the generalization performance that we are most concerned about. Figures 5 and 6 show exactly how the generalization performance varies with the number of hidden units. Above about 80 hidden units we found no significant difference in generalization performance as we increased the number of units. As we decrease the number of units below 80, the performance deteriorates significantly. A reasonably safe compromise was therefore judged to be 120 units.

Figure 7 shows the learning curves for a typical run of the network with 120 hidden units, a window size of 13 characters, *errcrit* = 0.01 and a single context flag to resolve the homograph ambiguities. The differences that we find in the learning rates for the regular and exception words in our model are in general agreement with human development (e.g. Backman et al., 1984). The network eventually achieved 100% performance on the training data and for non-words plateaued at 94.0% (comprised of 95.3% for regulars, 93.0% for exceptions and 93.8% for controls). Comparisons with other models are complicated by different authors using different non-word sets and scoring criteria, so bearing this in mind, the Seidenberg & McClelland (1989) model achieved 97.3% on the training data and about 65% on the Glushko non-words, Plaut et al. (1992) achieved 99.9% and 97.7% and Coltheart et al. (1993) achieved about 77% and 98%. For human subjects we would typically have virtually 100% on mono-syllabic words under normal conditions and about 95% on

the Glushko non-words under time pressure (Glushko, 1979).

Although our networks generally performed fairly well, and many of the non-word errors would be acceptable under more generous criteria of acceptability (e.g. 'wuff' → /wuf/ and 'wosh' → /w*S/ are counted as wrong by the above rules), there still remain a few errors of the kind humans would never make (e.g. 'zute' → /myt/). Sejnowski & Rosenberg (1987) note that they got a better generalization performance for the original NETtalk by using two hidden layers instead of only one. Unfortunately, our limited computational resources have prevented us from confirming this for our model. In the next section, however, we will examine the effect of a number of other variations of the basic model which we might expect to improve its performance.

It is also important for us to examine the effects of varying the over-learning parameter *errcrit*. Figure 8 shows that the training performance actually shows very little variation with *errcrit*. For very large *errcrit* (i.e. of order 1.0) the number of epoch required to reach a given level of performance is increased, but this is more than compensated for by the reduced number of patterns being trained on per epoch. Figure 9 indicates that there is also no significant variation in the final generalization performance as we vary *errcrit*. The temptation is therefore to keep *errcrit* quite high, but we shall see later that *errcrit* does actually have an important effect on the networks' resilience to damage.

As we find variations in performance from one human to another, we also find differences between networks with different parameters or even just different random initial weights prior to training. Of particular interest is the fact that different networks make their errors on different non-words, since this can give us valuable clues as to which non-words are giving real problems to the networks and which errors are simply due to random fluctuations. Table 1 shows the most common final non-word errors out of nine network runs that achieved perfect training performance (the four runs of Figures 8 & 9 plus the six successful runs of Figures 4 & 5 which includes one overlap). We see that there were only eight of the 166 non-words that were pronounced wrongly by more than one third of the networks and there is a clear reason for most of them. These reasons will prove crucial for determining the required improvements for this model in the next two sections.

Learning Trajectories

Since reading aloud has a particularly rich rule structure it is instructive to look more closely at the learning trajectories. The most common phoneme target is the blank, so the network invariably starts off by learning to output a blank for everything and consequently produces no pronunciation at all. It then begins to recognise that certain phonemes (usually /t/, /s/ and /r/) are quite common and regular and produces long single phoneme strings of these quite indiscriminately between long strings of blanks. At this stage all the output activations are still small (i.e. less than 0.5). Further through the first epoch it acquires stronger output activations for the more common phonemes (again /t/, /s/ and /r/) and these are generally occurring in the right places. There are also many strong blanks by now but these are less frequently in the right positions. The weaker outputs have become more varied but are predominantly the common consonants (/s/, /t/, /r/, /l/, /n/, /k/ and /p/) with occasional vowels (/a/, /A/ and /E/). By the end of the

first epoch (which uses about 730 of the training patterns) the network is already getting 1.4% of the full training set correct, but none of them with strong outputs. During the second and third epochs the blanks are more frequently occurring in the right places and more phonemes are being output strongly, including the first of the more complicated cases ('ea' → /E/ and 'sh' → /S/). The strong and correct outputs are beginning to line up the letters and phonemes quite strongly by now and many of the potential targets have been ruled out completely. We now have 33.6% of the training words and 34.9% of the non-words correct, many with strong outputs on each phoneme. By epoch eight, even the less common phonemes are being output strongly and the performance has increased to 67.4% on the training data (85.4% regulars, 8.3% exceptions) and 75.3% on the non-words. Most of the regular rules and major sub-rules have been learnt by epoch sixteen and we have 80.9% on the training data (95.8% regulars, 14.6% exceptions) and 91.6% on the non-words. The networks generalization ability (i.e. its performance on the non-words) reaches a peak after about thirty epochs and the remaining epochs (typically another 500) are used to set up the special purpose rules that are required to cope with the exceptional words but which tend to interfere with the generalization. The fact that, in order to deal with the exception words, we need to train well beyond the point where the generalization performance begins to decrease again suggests that learning here with conventional ambiguous (i.e. non-multi-target) training data is simply not feasible (Bullinaria, 1993b).

Letter-Phoneme Alignment

One thing that will obviously have a great effect on the networks' performance is how well they have managed to align the letters and phonemes correctly. If we compare the alignments of our nine networks in pairs we find that the mean number of training data words aligned differently is 939 (s.d. 387), i.e. over 30%. Some of these differences are due to clear mis-alignments (e.g. 'back' → /b–ak/) and it is reassuring to see that the networks can perform so well despite these errors. However, not all the differences are necessarily errors. Although we tend to talk about 'solving the alignment problem' there is actually no unique correct way to align all the letters and phonemes. For example, we can equally well have 'ng' → /N–/ or /–N/ and similarly 'th' → /D–/ or /–D/. In other cases there can simply be a preferred alignment out of two possibilities: for example, we could have 'ai' → /A–/ or /–A/, but since we often have 'a' → /A/ (e.g. in 'ace' → /As–/), the balance is tipped towards /A–/.

With the complication of equivalent alignment possibilities and only nine networks to compare, it is difficult to be sure, but the preliminary indication is that the more hidden units we use, the better the overall alignment we get. This is the impression we also obtain from a range of smaller scale multi-target problems (Bullinaria, 1993b).

Another way that we could improve the alignments would be to train the networks first on a subset of easy words (i.e. words that only have one possible alignment) before going on to learn the more difficult words. This is, in effect, what we do with children that are learning to read. We will avoid this course of action for the present, however, since it would inevitably lead to unnecessary criticism of the model.

Variations on the Basic Model

We saw in the preceding section that our basic connectionist model of reading already performs rather well. In this section we consider a number of variations on the basic model that we might expect to make it even more realistic.

No Explicit Blanks

Up till now we have had a separate output unit corresponding to a blank (i.e. no phoneme). An alternative, that might be deemed more realistic, is to have no explicit 'blank unit' but simply to have some threshold such that if no output activation exceeds that threshold then the network output is deemed to be a blank. Figure 10 shows the performance of such a variation compared with the basic model (using the same network and training parameters) shown in Figure 7. The performance on the training data is not significantly different but the network with explicit blanks performs significantly better on the non-words. Most of the differences can be identified as corresponding to cases where all the output activations are very low. In the basic model, as long as the most activated output is not the 'blank output', then that phoneme is deemed to be output however low its activation is. In the variation, the same level of activation may fail to reach the threshold and thus be deemed a blank output. Virtually all the extra non-word errors in the variation are caused by letters failing to reach the threshold in this way. Lowering *errcrit* so that all the errors are reduced was found to be insufficient to remedy the situation. It may be that introducing basins of attraction for the outputs and/or using larger training data sets may improve things but, for the present study, we are forced to persevere with explicit blanks. (This variation has been discussed in more detail in Bullinaria, 1993a.)

Multiple Phoneme Outputs

We have already noted that certain phoneme combinations (namely /dZ/ and /ks/) must be recoded for the basic model so that each word has less phonemes than letters. This is not a satisfactory state of affairs and does not account for other combinations of phonemes that naturally correspond to a single letter (for example, 'u' → /yU/ as in 'cube' → /kyUb/) which we can see from Table 1 cause many problems with the generalization. The obvious way to proceed here is to allow more than one output phoneme per presentation of each word.

Let us first consider the case with two output phonemes (as shown in Figure 11). Setting this up for the original NETtalk with its pre-aligned training data would be straightforward. The main problem we have here with our multi-target training data is that the number of possible output targets for each word grows rather quickly with the number of output phonemes. For our previous example of the word 'ace' → /As/, we now have fifteen targets (As — —, A– s– —, A– –s —, etc.) compared with only three (A s –, A – s, – A s) when we had a single phoneme output. The word 'cube' → /kyUb/ now has 105 targets compared with only one before. One way to restrict the number of targets without making any assumptions about the nature of the training data is for each presentation to have each phoneme pair left justified, i.e. to allow blank outputs only to the right of any phonemes, so that /s–/ is allowed but /–s/ is not. For 'ace' we then have only six

targets (As — —, A– s– —, A– — s–, — As —, — A– s–, — — As) and for 'cube' we are left with only nineteen targets.

From a computational point of view the number of targets is not very important compared with, for example, the number of hidden or output units. However many targets we have, there is still only one forward pass through the network and one backward pass. The difference is only in the number of targets that need to be compared with the actual network output and (assuming the whole set of targets is stored in memory rather than repeatedly recalculated) the time taken to do this will usually be very small in comparison with the other computations required. Exactly how this process might be carried out in real brains is not clear at present but, given human pattern matching ability in other areas, it should not pose too much of a problem.

In cases where the target comparison time does become significant, there is a computational trick (which was *not* actually used for any of the simulations described in this paper) that can speed up the process. As noted in our discussion of the learning trajectories, most of the alignments are figured out in the first few epochs so, if the best target for each word can be remembered from one epoch to the next (which is easily arranged), the number of target tests tends to one per word very quickly. This is because we can test that previous best target first and we know that, once we have found a target with a total error score less than 1.0, all the other targets necessarily have a greater error score. It should be emphasised again that such computational trickery merely speeds up the computation, it does not change the effect of the learning algorithm.

A number of simulations of these two output phoneme networks were carried out and, despite the much greater number of targets, the networks still managed to learn to use a sensible target for each word. In practice, the opportunity for the network to output two phonemes for a single word presentation was actually used very rarely. The most common instances were the 'j' → /dZ/, 'g' → /dZ/ and 'x' → /ks/ cases mentioned previously. Apart from these and a few exception words (such as 'choir' → /k– — wI — r–/ and 'once' → /w^ n– s– —/), the only other uses of the second phoneme were with 'u' → /yU/ and 'ch' → /tS/ but here the first phoneme (i.e. the /y/ and /t/) was always associated with the previous presentation (e.g. so that 'cube' → /ky U– b– —/). That the networks made such sensible use of their new degree of freedom is encouraging and clearly beneficial to the generalization performances.

The use of more than two output phonemes per presentation is a straightforward extension of this, though the number of targets per input can grow rather large and the extra outputs are rarely used. Of particular interest is the case of four sets of output units since this is sufficient for us to be able to turn the training data around and run the system as a model of spelling with the pronunciations on the inputs and the corresponding letter strings on the outputs. Although spelling is clearly closely related to reading aloud it is somewhat more than a simple inverse mapping (e.g. Frith, 1980; Kreiner & Gough, 1990). The rule structures are considerably more ambiguous and there are over 440 homophones in the Seidenberg & McClelland data set alone. Consequently, we shall leave the analysis of these models of spelling for another paper.

Beginning and End of Word Markers

The basic model has no explicit markers for the beginnings and ends of words and it

is reasonable to expect that such markers would facilitate the learning of certain rules. The fact that the network often pronounces the Glushko non-word 'mone' as /mw^n/ by analogy with 'one'→ /w^n/ rather than the regular /mOn/ is evidence of this (see Table 1). It is easy to incorporate such markers into the training data by simply appending the appropriate extra characters to the beginnings and ends of each letter and phoneme string. However, there are many possible variations on this theme which should be investigated for use in more sophisticated models of this type. For the letters we can have different markers for the beginning and ends of words or we can have a single 'word separation' character. For the phonemes we can output distinct word separation characters, or our existing no pronunciation 'blanks', or some combination of these. Unfortunately, the facts that the model is already so successful without these explicit markers and that there are variations between different runs anyway, mean that it is difficult to confirm any significant differences in performance between the different types of markers. What is clear however, is that when they are included, the markers *are* able to rectify the problems experienced with non-words such as 'mone' and 'wone'.

In real speech the gaps in the pronunciations are often within words rather than in between them and this leads us on to the whole problem of stress in multi-syllabic words and the relationships with adjacent words and semantics. In the multiple-phoneme output models there is room for the inclusion of stress and other markers in the output strings and hence we have the possibility of working with continuous stressed speech. Once again however, this is beyond the scope of this paper and will be reported elsewhere.

No Hidden Units

For comparison with other types of model which do not perform as well as our basic neural network, such as analogy models which find it difficult to get as much as 80% generalization performance (e.g. Sullivan & Damper, 1992), it is interesting to see how well our model can do without any hidden units, i.e. with only direct input to output connections. It is not surprising that these networks cannot perform as well as the models that do have hidden units, since it is easy for us to construct non-linearly separable exception words which are well known to require hidden units (e.g. Minsky & Papert, 1969). What might be considered surprising is how well they can do. Since some output errors will remain however long we train for and the training regime presents different words in different orders in each epoch, the weights never fully stabilise and there remain fluctuations in the performance. Figure 12 shows the learning curves for the two phoneme output case with the performances averaged over several epochs for the data points beyond epoch 512. This network reached a final average performance of 88.5% on the Seidenberg & McClelland training data (comprised of 100.0% for regulars, 31.2% for exceptions) and 90.4% on the non-words. Considering how well such a simple association can do, we should perhaps investigate more closely what exactly is happening in other models (such as statistical analogy models) which generally do worse than this. Once again, the considerably superior performance on the regular words compared with the exception words is reminiscent of some forms of developmental dyslexia and will be discussed in more detail in the section devoted to learning and developmental dyslexia.

Alternative Training Data

The Seidenberg & McClelland (1989) set of mono-syllabic words has become the standard for training models of reading and for the purposes of comparing models it would be foolish not to use it. However, it has been found that if smaller training sets are used, then obviously incorrect grapheme-phoneme correspondences can be learnt by our models (e.g. 'ace' → /–As/ instead of /As–/) without noticeably affecting the output performance on that training set. Moreover, it is clear that the more training examples we have, the easier it will be for the network to identify the precise extent of the various sub-rules. It is likely, therefore, that simply using larger training sets could further improve our network's generalization performance, particularly since more words here means longer words and these will tend to be more regular than the words we have used thus far. It is also important for us to test that our model really can handle large numbers of poly-syllabic words in its training data.

We therefore constructed a suitable set of training data consisting of words derived from the Lund corpus (Svartvik & Quirk, 1980). To simplify the performance comparisons, the data set was constructed with the single phoneme output model in mind. We started with a computerized lexicon containing 61141 words and their corresponding phoneme strings (using 44 different phonemes). Of these words, 2415 had more phonemes than letters and most of these could be identified as arising from nine correspondences of the form 'x' → /ks/, 'u' → /yU/, etc. Since these would also cause alignment problems with many words that did have fewer phonemes than letters, these nine phoneme pairs were replaced throughout by new 'phoneme' symbols brining the total number of phonemes up to 53. The remaining 387 'words' with more phonemes than letters (such as 'mrs' → /misiz/) were removed, as were 524 words with less than 2 letters or more than 14 letters. We will refer to this remainder as the 'main lexicon'. Of these there were 14014 words that had a non-zero frequency count in the Lund corpus. To simplify the computer simulation (i.e. to be able to run it on a computer with limited memory and computational power), words that would have had more than 64 training targets were removed leaving 12840 words of which none were homographs. The distribution of the logarithms of the Lund frequency counts was similar to that of the Seidenberg & McClelland corpus. For convenience when comparing results, the frequency counts of the words contained in the Seidenberg & McClelland corpus were replaced by the Kucera & Francis counts and words that occurred in the Seidenberg & McClelland corpus and the 'main lexicon' but not the reduced 12840 word set were added to that set. Three words in the Seidenberg & McClelland set (namely 'frappe', 'math' and 'plow') that are used in various experiments but were missing from the 'main lexicon' were also added to our word set and the least regular of each of the remaining homographs were removed giving us a final lexicon of 13891 words.

This lexicon was then used to train a range of networks. A few trial runs revealed that we needed to increase the window size from 13 to 17 characters to deal with all the long range dependencies. One main run was then carried out with 160 hidden units, an *errcrit* of 0.01 and explicit beginning and end of word markers. The learning curves are shown in Figure 13. The increased number of words used per epoch gives the impression that learning is faster than in the basic model (shown in Figure 7) but the total number of epochs required to deal with all the exception

words was actually increased. This, together with the increased lengths of the words, meant that the overall training time was increased by a factor of more than six. (Unfortunately, a few acronyms such as 'iou' and foreign words such as 'famille' slipped through the word selection procedure and caused particular problems with the training.) As hoped, the non-word performance was improved (to 97.0%), but direct comparison with the basic model is complicated by the different phonetic transcriptions used in the larger training set. (It is unlikely that a small number of homographs would make much difference to the performance of a network trained on such a large data set.)

Finally, we note that (as for the original NETtalk) it is a basically straightforward extension to train our model on a continuous stream of words and on transcribed real informal speech containing errors and inconsistencies. A full investigation of this is planned and will be reported elsewhere.

Recurrent Connections

Given that the phoneme (or stress marker) to be output next by the network will often have a strong dependence on the previous phonemes (or stress markers) it is quite feasible that the models' performance could be further improved by the incorporation of recurrent connections. So far we have only implemented the simplest form of recurrent connections whereby a copy of the previous output is included as an extra input into the network (as shown in Figure 14). Various other types of recurrent connections are possible, such as Elman (1990) type context units and basins of attraction (Plaut & Shallice, 1992), and it may be necessary to incorporate them in the future to account for other aspects of the reading process or simply to improve the existing performance. In fact, we shall suggest in the concluding section that the whole moving window architecture should be replaced by an equivalent system of recurrent connections.

On the other hand, rather than improving our model, such recurrent connections could conceivably confuse the multi-target learning algorithm and prevent the network from settling down into using a sensible set of targets. Either way, the effects of such connections were in need of investigation. Consequently, three recurrent networks with 40, 80 and 120 hidden units, *errcrit* = 0.01 and a window size of 13 characters were trained on the Seidenberg & McClelland training data. As with our other variations on the basic model, it was difficult to distinguish real changes in performance over the corresponding non-recurrent networks from statistical fluctuations. The recurrent versions learnt the training data perfectly in a similar number of epochs to their non-recurrent counterparts and Table 3 shows that they also have a similar (and possibly slightly better) performance on non-words. The important point is that these extra connections do not degrade the networks' performance and do not disrupt the multi-target learning.

Parallel Implementations

Our neural network models are very computationally intensive and one way to speed up the training is to perform the computations in parallel. The learning algorithm was therefore re-implemented in C* for the Connection Machine . (C* and Connection Machine are registered trademarks of Thinking Machines Corporation.)

The simplest implementation involved distributing the training patterns over

the computer's processors. This necessitated the use of batch training rather than on-line training and unfortunately, as a consequence, the number of epochs required to reach 100% performance on the training data increased so much that despite the considerably faster computer we never managed to complete a single run. The problem was that in order to perform a reasonable approximation to gradient descent and to settle into a good set of targets the total step size in weight space must be kept small. If our batch size is 8192 patterns (corresponding to the number of processors on half of our local Connection Machine) then to keep the same total step size we must reduce our back-propagation learning rate by some appropriate factor from that used by the on-line version. Given that we keep the step sizes reasonably optimal in both cases, we then have a race between single accurate steps for the batch mode against 8192 inaccurate steps for the on-line mode. In this case the on-line version won by a long way and we went back to our old serial computers. It is possible that varying the step size during the learning will improve matters, but this is likely to involve a great deal of trial and error and has not yet been carried out.

There are other (though not so straightforward) ways to slice up the calculations between the computer's processors, such as having the processors acting as network units or connections (e.g. Deprit, 1989). This will not require batch processing and is more like the kind of parallel processing that happens in real brains. Such approaches are obviously worth further consideration in the future.

Performance of the Best Model

In this section we describe in detail the performance of our best reading model to date, and compare it with normal human performance.

The variations to our basic model that were discussed in the last section can clearly be applied in various different combinations. The changes that appeared to improve performance were multiple phoneme outputs, beginning and end of word markers, recurrent connections and larger training data sets. The first two of these seem to be necessary to remedy several of the most common non-word errors of Table 1 and so we will definitely include these in our improved model. However, in order to ease the comparison with other models and to keep the model as simple as possible (not to mention reducing the strain on our computational resources) we will forgo, for now, the use of recurrent connections and larger training sets.

The best combination model tried so far therefore has explicit blanks in the output units, two output phonemes per presentation, explicit beginning and end of word markers, no recurrent connections, on-line training with the Seidenberg & McClelland training corpus, a window size of 13 characters and a single context flag to resolve the homograph ambiguities. We had very little information to help us choose the optimal number of hidden units and value of *errcrit*. Experiments with very small scale models of this type (Bullinaria, 1993b) suggest that the probability of achieving a good generalization performance increases with the number of hidden units. Moreover, the more hidden units we have, the more brain-like the system is likely to be and the more likely it is to behave reasonably realistically when damaged. A preliminary investigation of network damage also indicated that reducing *errcrit* leads to more resilience and more realistic post-damage performance. Thus, bearing in mind our limited computational resources, we

carried out two runs with 300 hidden units, one with *errcrit* = 0.0001 and one with *errcrit* = 0.

The two sets of learning curves obtained were very similar and those of the *errcrit* = 0 network are shown in Figure 15. We see that the curves are also very similar to those of the basic model (shown in Figure 7), except that the final non-word performance is now slightly better. The learning trajectories are closely related to those of the basic model. The network starts off by outputting blanks for everything. Then, since the phonemes (as distinct from blanks) are much more common targets on the first set of output units, it proceeds as though the second set were not there and goes through the same learning process as before. Only once the letters and phonemes are strongly lined up is it forced to begin making use of the second set of outputs and we end up with the pattern of alignments discussed previously. We will now look at various aspects of the models' performance in more detail.

Generalization

Table 3 summarizes the two models' generalization (i.e. non-word) performance. For each case we show the total percentages of the three sets of non-words that were pronounced acceptably together with the total percentages that were pronounced regularly. Also shown is the percentage of the output pronunciations that have a close rival output. Not surprisingly, the non-regular pronunciations and the close rival outputs occur much more frequently for the exceptional non-words (i.e. non-words that are derived from exceptional base words) than for the regular non-words. The 'Overlap' sub-table counts the instances that co-occur for both networks and consequently provides an indication of the variations between the two networks. (The only two non-word errors that were made by both networks were 'wosh' → /wuS/ and 'lokes' → /lOkz/.) Finally, we see that the networks actually perform *better* than humans working under time constraints (Glushko, 1979). That humans often pronounce non-words by analogy with similar exception words, rather than by using the most regular rules, has often been taken to be evidence against the existence of a traditional GPC route in a dual route model. We can see from Table 3 that our models are actually *more* likely to pronounce non-words by analogy than humans.

Additional evidence of the models operating by analogy comes from the way that the networks learn to align the letters and phonemes. It is interesting that, unlike the smaller and simpler networks considered in our discussion of the basic model, the two networks here chose to solve the alignment problem in a remarkably similar manner: there were only 22 different alignments out of the 2998 words. Where one of the two versions could be judged superior, the *errcrit* = 0 network won by 10 items to 1, which is consistent with its slightly better performance on the non-words. Although the regular alignments would have worked just as well (as in all the other networks we have tested), whenever an 'l' follows a double vowel, the 'l' phoneme always takes the central position and the 'blank' the final position (e.g. 'ail' → /Al–/ rather than /A–l/). Such apparently unnecessary sub-rules do not seem to affect the generalization performance. Similarly unnecessary sub-rules corresponding to exception word analogies are also likely to occur. For the networks to operate efficiently, the rules and sub-rules clearly need to be arranged into a context dependent hierarchy, but there will not necessarily be a unique

hierarchy to account for a given set of training data. For example, consider the 17 words in the training data ending in 'ead'. Six have the regular /Ed/ ending whilst eleven have the irregular /ed/ ending. The /ed/ sub-rule must take precedence over the regular rule in a particular set of contexts which must include all of the eleven words but none of the six. Which rule has precedence in other cases, e.g. when we test the network's generalization performance on non-words, is not determined. This is why our networks and humans sometimes respond with a strictly regular pronunciation and sometimes by analogy with an irregular word. (Humans also suffer from priming effects that do not exist in our current models.) It is only by using a more representative training set that such rule ambiguities can be resolved. Since our networks are already trained on virtually all mono-syllabic words any enlargement of the training data set will primarily consist of longer and poly-syllabic words which tend to be more regular. Consequently, they will tend to restrict the sub-rule contexts more tightly than the mono-syllabic words used thus far and hence, as we employ more realistic training data, the generalization performance of our models should become more regular and human like.

Another possible reason why humans may give more regular non-word responses than our networks is that the GPC route in humans never achieves perfect performance on the training data and leaves a separate semantic/lexical route to deal with most of the exception words. This is equivalent to stopping the training early (e.g. near the generalization peak) or using some other regularization device that restricts the network's acquisition of the exception words (as discussed further in the section on damage and acquired dyslexia). Either way, because the exception word performance remains relatively low, the non-word pronunciations are less likely to be formed by analogy with the exception words and are hence more likely to be regular. This is confirmed by the results in Table 4 which shows our two networks' generalization performance after 32 epochs where the exception word performance is still only about 30%. The mean proportion of acceptable exception non-word pronunciations that are regular is now 83.6% compared with 64.0% after full training and 81.6% in humans. These results also give an indication of the likely performance when the network has acquired a smaller proportion of exception words for other reasons, e.g. because it has been trained on a more realistic set of training data.

An interesting study by Treiman & Zukowski (1988) suggests that when humans pronounce non-words by analogy (rather than according to the main GPC rules), they use vowel plus final consonant units to a greater degree than initial vowel plus consonant units or vowels alone. They arrived at this conclusion by studying the pronunciation of 36 non-words derived from the irregular base words 'friend', 'said', 'been' and 'gone' by changing the initial consonant cluster, the final consonant cluster or both consonant clusters. The same 36 non-words were tried on our two networks. All 72 responses were acceptable with 6 of them pronounced by analogy. Unfortunately, none of the analogies employed by our networks corresponded to the base words in the experiment, so no conclusions could be drawn. Since we only have two networks compared with ten subjects in the experiment, this is not a disastrous result. It is clearly something that is worth investigating further when more trained networks become available. (Since all the base words are fairly common it is also quite conceivable that the effect in humans is enhanced by priming effects not yet possible in our networks.)

Finally for this section, we compared the performance of our two networks

with a comparable network trained on pre-processed training data, i.e. data for which the alignment problem had already been solved. Since the final set of alignments learnt by the *errcrit* = 0 network was essentially perfect (modulo the 'ail' anomaly mentioned already and which did not result in any of the non-word errors), we used the training data with these alignments built in by hand to train a network with exactly the same parameters. The learning curves are too similar to those of the non-aligned case (shown in Figure 15) to be worth presenting here. As noted previously, the multi-target networks have generally learnt the appropriate alignments by epoch eight. Before epoch eight there are two opposing effects that may be significant. Firstly, the fact that no training at all is being done on incorrect targets means that learning should be speeded up. Secondly, the fact that there is only one possible target for each word means that there is less chance of arriving at an acceptable target 'by accident'. Since progress in the first few epochs depends so crucially on the random order of the training data presentation anyway, it is virtually impossible to determine what is actually happening. The important observation is that beyond epoch eight there is no significant difference at all between the aligned and non-aligned cases in either training or generalization performance.

Extracting the Rules

Our network's performance on the non-words demonstrates quite clearly that it has learnt a good set of GPC rules. Unlike other models that employ complicated representations such as Wickelfeatures, it is easy to for us to see exactly what output is produced by different combinations of input letters. First, Table 5 shows the default outputs for each letter in the alphabet and the corresponding error scores. Most of the consonants give rise to the expected phonemes. The vowels, on the other hand, either have high errors or give the default blank output. This is because the vowel pronunciations depend so much more critically on the context. To get the response equivalent to asking a human to pronounce a single letter on its own we need to sandwich that letter between beginning and end of word markers. As expected, this does give strong non-blank outputs for all the letters including the vowels. In networks that do not have beginning and end of word markers, such as our original basic model, the default outputs are strong non-blank phonemes for the letters by themselves.

We can simulate the effect of a generic consonant (which we shall represent by 'C') by activating the input units corresponding to the ten strongest consonants (namely d, f, k, l, m, n, p, r, s, t) at one tenth of their full value. The total input is then equivalent to a single consonant but all the corresponding output activations are less than 0.02. Using this and the beginning and end of word marker '|' we can examine the rules learnt for the vowels in different contexts.

Table 6 shows how the long and short vowels are produced in different contexts, in particular the rule that a final 'e' lengthens the preceding vowel. For the double vowels there is somewhat less ambiguity when they are on their own, as we can see from Table 7. In Table 8 we have the outputs when they are sandwiched between generic consonants.

These results constitute the main GPC rules, but they can be over-ridden by more powerful sub-rules. Table 9 gives some well known consonant examples. First, the letter 'c' is pronounced differently (i.e. hard or soft) depending on which

vowel follows it. Next we see some of the familiar pairs of letters that have special pronunciations. Finally, we see that some letters and combinations are pronounced differently at the beginning of words. Table 10 shows some typical vowel sub-rules. For example, we see how the 'oo' → /U/ rule is over-ridden if the 'oo' is followed by a 'k' but not a 'ke'. Similar special rules apply for the vowels in the combinations 'ur', 'or', 'ol and 'oth'.

In Table 11 we can begin to see how the exception words are dealt with. The 'i' in 'pint' is pronounced /I/ instead of the regular /i/ and the 'i' in 'give' is pronounced /i/ instead of the regular /I/. We see that as we supply more and more context information the error measure increases until the new sub-rule takes over completely. In this way, we can also determine which context information is most useful in defining each sub-rule.

By plotting (in Figure 16) the mean absolute weight values for connections coming out of the various input window positions we have further evidence that the network is using the context information in a sensible manner with the centre of the window having more effect than the extremities. If the curve was purely the result of the way that the words are presented (i.e. with every presentation having a letter in the central position and letters in positions 1 and 13 being relatively rare) it would be symmetric around the central position 7. In fact it is skewed to the right in the same way as the actual mutual information provided by the neighbouring letters in the training data (Lucassen & Mercer, 1984).

It has been suggested (e.g. Glushko, 1979) that, when humans pronounce new words or non-words, they do not do so by using a set of GPC rules, but rather, they operate by analogy with words that they already know. So we have to ask: Has our network really learnt a set of rules, or is it merely operating by analogy with the words it has been trained on, or is the distinction meaningless anyway? From Table 9 we see, for example, that it knows the 'rule' that 'ph' is pronounced /f/, but it is equally valid to say that the 'ph' is pronounced by analogy with the 'ph''s occurring in the training data. More revealing perhaps, is a consideration of the well known rule that a terminal 'e' in a word lengthens the preceding vowel. Table 6 suggests that the network knows this rule for all the vowels, but it is not clear from this that it really is able to apply this rule to cases (i.e. vowel-consonant clusters) not present in the training data. To test this point we investigated the network's performance on word strings of the form '*VC*e|' (where *V* is any vowel and *C* is any consonant) which do not occur in the training data. There are actually only 12 *VC*'s of this form that are not in the Seidenberg & McClelland corpus (namely: eb, oc, ef, of, uf, og, ek, el, ex, ix, ox, ux) of which we reject two (namely: of, og) because their *VC*| follow a sub-rule rather than the main rule and to which we add one (namely: ax) whose *VC*e| form follows a sub-rule rather than the main rule. For each of these we tested the networks' output on both '|*KVC*e|' and '|*KVC* |' where *K* is one of a set 39 common initial consonant clusters (namely: b, c, d, f, g, h, j, k, l, m, n, p, r, s, t, v, w, y, z, qu, ph, bl, br, ch, cl, cr, fl, gr, sh, sl, sp, st, th, tr, shr, spr, squ, str, thr) or the generic consonant C. Table 12 shows the results averaged over our two networks. Both networks have problems with the 'el' case. For the other 10 cases, both networks have perfect performance on all the short vowels and on the long vowels with the generic consonant. The performances on the long vowels with the full set of initial consonants are good, but by no means perfect. However, we do not really expect perfect performance on the final 'e' lengthening the vowel because there are numerous exceptions to the rule in the training data (e.g. 'were' → /wer/ and 'none'

→ /n^n/). As discussed in the previous section on generalization, we can expect the performance to improve, i.e. become more regular, if we train on a more realistic set of training data. Although the 'e rule' is a classic example of a GPC rule, even here the distinction between rule and analogy is unclear - one could still argue that the network is simply operating by analogy with '*V*_e' independent of the consonant that fills the blank. If one takes this point of view, then the network does not need to learn any rules - it (and humans) can read everything purely by analogy. This is not to say that abstraction does not take place - the network (and humans) still need to learn which of the possible sub-words are most important when pronouncing by analogy (e.g. '*V*_e|' rather than '|*KV*').

Lexical Decision

Any complete model of reading should be able to distinguish between words it has been trained on and words or non-words that it has not seen before, i.e. it should be able to perform lexical decision. Experiments involving lexical decision are generally regarded as providing important clues concerning the processes underlying reading (e.g. Andrews, 1982; Seidenberg et al., 1984; Waters & Seidenberg, 1985). However, the only decision criterion possible in the current model is provided by the mean square output activation errors. Figure 17 shows the distribution of error scores of the network with *errcrit* = 0.0001 for the training data set and the three sets of non-words. A similar graph is obtained for the *errcrit* = 0 case. It is clear that, although the mean error score for the training words is somewhat less than that for the non-words, there is considerable overlap between the training words and the non-words and hence the model is unable to perform lexical decision in this way.

Unlike our model, the Seidenberg & McClelland model had separate sets of phonological and orthographic output units and it was suggested that the orthographic errors could be used as a measure of familiarity (Seidenberg & McClelland, 1989). Unfortunately, it was later shown that this does not work either (Besner et al., 1990). It seems that a separate lexical/semantic system will be necessary in order to account for lexical decision.

Interference Effects

Seidenberg & McClelland (1989) illustrated quite well the effect that training on one word had on the performance on another by considering how the performance of their trained network on the regular word 'tint' varied as further training took place. They showed that additional training with the regular word 'mint' improved performance, whilst training with the irregular word 'pint' worsened performance and training with the control words 'rasp' and 'tent' had relatively little effect.

Naturally, since the same basic learning algorithm is involved here, we should expect to be able to proceed similarly with our networks. The problem with our model is that its error scores for the regular words 'mint', 'rasp' and 'tent' are already so low that further training has virtually no effect on the network's weights and hence has little effect on the performance on 'tint'. For the comparison of effects to be valid we must train on words with well matched and relatively high error scores. Figure 18 shows that the effect on the word 'tint' of training on the irregular

word ‘pint’ (error 0.00296) was worsened performance, the control word ‘comb’ (0.00296) had little effect and the regular word ‘dint’ (0.00252) improved performance. It also shows that training on the regular non-word ‘wint’, that already has a very low error (0.000001), has no effect on the word ‘tint’. Similarly, because of its relatively low error (0.00022), even training on ‘tint’ itself has less effect than ‘dint’. This confirms our natural intuitions about how the effects of the different training words reinforce and counteract each other during learning.

By considering the implications of this for a full set of training data, which by definition contains mostly regular words, it is easy to understand why the model tends to learn the regular words before the exception words and why it generally performs better (i.e. has lower output activation error scores) on regular words than on exception words. It also implies that high frequency exception words should be learnt faster than low frequency exception words and that we should expect ceiling effects whereby the performance on the higher frequency exception words eventually catches up that of the regular words. We will now see that these simple observations can go a long way to explaining the results of numerous naming latency experiments.

Naming Latencies

It has been suggested (e.g. Seidenberg & McClelland, 1989; Cohen, Dunbar & McClelland, 1990) that neural network output activation error scores are correlated with response times under pressure. The idea is that in more realistic cascaded processing systems in which the activations build up to some threshold over time, the higher the outputs (i.e. the lower the errors), the lower the time taken to reach the threshold (McClelland, 1979; McClelland & Rumelhart, 1981; Norris, 1993). If this is correct and we can consider our simple feedforward networks to be a reasonable approximation to these cascaded systems, then it is appropriate to compare the outputs of our model with the results of numerous naming latency experiments that exist in the literature (for example, that words are generally pronounced more quickly the more regular they are and the higher frequency they are).

Unfortunately, these naming latency experiments are notoriously difficult. Different experimental conditions often result in a wide variation in naming latencies for the same words (e.g. Waters & Seidenberg, 1986, experiments 1 and 2) and there seems to be no simple scale factor between the different set-ups. There is also a wide variation between individual words and a large overlap in scores between the various word types (e.g. Waters & Seidenberg, 1986, Appendix A). Nevertheless, many interesting and statistically significant results have been obtained. The problem is that it is often difficult to justify general claims about what exactly is causing the effects. By comparing these experimental results with the corresponding results from our model, which (by construction) can only capture the effects of a subset of the processes underlying reading, we can hope to throw further light on the matter.

There are many problems with relating the models’ error scores to the naming latencies. Firstly, the way our network model works doesn’t really correspond to any particular experimental set-up (e.g. it has no time constraints, the model never makes errors once it has been trained, etc.). Seidenberg & McClelland (1989) were consequently forced to use a whole series of different linear relations between the

error scores in their model and the naming latencies to account for the different experimental results. Secondly, because of the random nature of the training data presentation, some anomalously high or low error scores can arise simply due to the presence or absence of particular training patterns within the last few word presentations or epochs. These priming (or anti-priming) effects are particularly problematic for the low frequency words, which is where most of the significant effects occur. Moreover, during training we only have a total of about 400000 word presentations and the logarithmic relation between the actual word frequencies and the models' presentation frequency is likely to diminish many of the frequency effects found in humans. Unfortunately, the computer time required to do otherwise is prohibitive.

Perhaps the most fundamental problem we face is to decide on the precise mathematical relation that should exist between the network output error scores and the simulated naming latencies. Seidenberg & McClelland (1989) used a simple linear transformation of the mean squared error. However, in our model the final error scores vary over several orders of magnitude, whereas in the experiments the latencies rarely vary by more than a factor of two. (The actual distribution of error scores for our model is shown on a logarithmic scale in Figure 17.) A simple linear relation will thus give a very skewed distribution that is not going to be particularly realistic. The obvious alternative is to use a logarithmic relation, which results in a more realistic (nearer Gaussian) distribution of errors, though it is difficult to justify this relation. (Remember, however, that one of the assumptions underlying the standard analysis of variance is that the distribution of observations on the dependent variable is near normal within each group.) If one argues that the naming latency should be given by the time taken for the output activation to build up to some threshold in a more realistic network architecture and that this is inversely related to the input into the largest output unit, then we can use the fact that $error = \exp(-|input|)$ for small errors to justify a negative inverse logarithmic relation between the errors and naming latencies. Perhaps even more justifiably, we should have the time proportional to the direct sum over input letters of the inverses of the inputs into the outputs. These relations also result in reasonable latency distributions when the errors are small, but do not make sense when the errors are large (e.g. with certain non-words) since the inputs are negative for output activations less than 0.5. One solution to this, that has been used before (Cleeremans & McClelland, 1991), is to normalise the outputs by taking their Luce ratios (Luce, 1963), i.e. dividing each output by the sum of all the outputs. These fractions of the total output give a fairer indication of which output unit is winning, but still require a logarithmic transformation to give anything like a normal distribution.

Given the problems noted above, it is clear that we will have to restrict ourselves to looking for general trends rather than expecting to find precise mappings between the models and particular experiments on humans. Unfortunately, even the main significant effects are not always independent of the precise mathematical relations and experiments used. Consequently, for each word type we will tabulate the simulated latencies for each of four mathematical relations (namely log error, negative inverse log error, sum inverse inputs and log Luce ratio). The linear scale factors used for these scores were chosen simply for clarity and are essentially arbitrary, but are consistent across all data sets. The statistical significances of the differences (using standard t-tests and analyses of variance) will also be tabulated in detail to give a clear indication of the subtle differences between

the different relations and word sets. There are also many other possible variations on our four relations, but the four will be sufficient to give a fair indication of the dependencies on the details. We will also restrict ourselves to actually plotting the results only for the simplest mean log error relation for each word type.

The three main word types we will consider are exception, regular inconsistent and strange. Each of these types will be compared with regular control words and with each other. The main differences in humans are found for the low frequency words so we will begin by plotting (in Figure 19) the evolution during training of the output activation error scores for the low frequency words of each type. This shows quite clearly the basic error score hierarchy which we can now refer to in a detailed examination of each word type. Although it is the final error scores that we are primarily interested in here, the learning curves provide important additional information. First, they indicate how the final errors arise as a result of the learning process and where ceiling effects are playing a crucial role. Secondly, they give an indication of the results we can expect if the training is stopped early, either due to developmental problems or because some other (e.g. semantic) route has learnt to take care of the remaining errors. We will end this section with a discussion of some of the problems involved in accounting for the naming latency results for unique words, non-words in general and pseudohomophones.

Exception Words.

Figures 20 and 21 show the learning and error curves for the exception words and regular control words of Taraban & McClelland (1987). We see that the regular words are acquired first and that it is only after most of them been learnt (around epoch 8) that the network starts to make significant progress with the exception words. Early on in training (e.g. in the first 64 epochs) the random fluctuations caused by the randomness of the word presentation order obscure the differences between the high and low frequency words but later we see clearly the superiority of the high frequency words. We can also see that by the end of training the differences between the high and low frequency words will be much less for the regular words than for the exception words.

In humans there is considerable experimental evidence that the pronunciation latencies are longer for exception words than for regular words and also that there are significant word frequency effects (e.g. Baron & Strawson, 1976; Glushko, 1979; Waters & Seidenberg, 1986; Taraban & McClelland, 1987). Figure 22 summarizes the results of the Waters & Seidenberg and Taraban & McClelland experiments. There are significant overall type and frequency effects. The type effect for the low frequency words was larger than that for the high frequency words. The frequency effect is significant for the exception words but not the controls. The two way interaction between type and frequency, however, was not significant.

It seems that there is general agreement between our model and what happens for humans, but there are some interesting differences. Figure 23 shows the mean logarithm of the error scores for our model, for the same sets of words as used for Figure 22. The full set of statistical results, shown in Table 13, is more informative. The results from the two word sets are in general agreement, with the larger data set sizes of Taraban & McClelland giving slightly more statistical significance than those of Waters & Seidenberg. There is a clear type effect for all cases. In most cases we have a significant frequency effect for the exception words but not the controls, resulting in a significant overall frequency effect. Only for the 'sum inverse inputs'

relation do we find no frequency effects at all. We find a significant interaction effect only for the 'inverse log error' relation.

The most obvious difference between our model and humans is the greatly reduced frequency effect. Given our understanding of the learning curves, it seems highly probable that it is merely our logarithmic reduction of the word frequencies in the training data (that we had to use to speed up the training to a feasible level) that has diminished most of the frequency effects. Clearly, for more realistic results we need to find some way to proceed with a more accurate representation of the word frequencies in the training data. The effect of this would be to reduce all the error scores on the high frequency words and hence bring our model's performance (Figure 23) into line with the experiments (Figure 22).

Regular Inconsistent Words.

It is clear from Figure 18 that the learning of the exception words in our model can have a negative effect on the learning of the similar regular words. In order to investigate such interference effects in humans, Glushko (1979) carried out experiments on a class of words (termed *regular inconsistent*) which are regular, in the sense of following the main GPC rules, but which are close neighbours of an exceptional word. A typical example is the word 'five' → /fIv/ which is regular but whose pronunciation may be confused or slowed by the nearby exception word 'give' → /giv/. Glushko found that regular inconsistent words do indeed exhibit longer naming latencies than consistent regular words and this has been interpreted as evidence against dual route models (e.g. Henderson, 1982). However, later studies, in particular that of Taraban & McClelland (1987), have failed to reproduce such effects. The results of these two studies are summarized in Figure 24. The original Glushko results were split into equal high and low frequency sets on the basis of the Kucera & Francis (1967) word frequencies used for our model.

Figures 25 and 26 show the learning and error curves during training of our model for the regular inconsistent words and corresponding control words of Taraban & McClelland (1987). Figure 25 shows quite clearly how the learning of the regular inconsistent words (particularly the low frequency ones) has been slowed compared with their consistent controls. This is also clear from the error scores shown in Figure 26. Comparison with Figures 19, 20 and 21 indicates that our model predicts that we should find similar naming latency type and frequency effects for regular inconsistent words as we get for exception words, but to a lesser degree.

It would seem, then, that our model agrees with Glushko's results rather than Taraban & McClelland's. The model's full set of simulated naming latencies and statistical significances are shown in Table 14. For the Taraban & McClelland word set, we find no significant effects at all for the 'sum inverse inputs' relation and significant type effects but no frequency or interaction effects for the other three relations. For the Glushko word set we find different patterns of effects for the different relations. We get significant type and frequency effects but no significant interaction effect for the 'sum inverse inputs' relation. For the 'log error' and 'log Luce ratio' relations we get significant type and interaction effects but no frequency effect. For the 'inverse log error' relation the frequency, type and interaction effects are all significant.

It is clear that, whilst the details of the mathematical relations between the network output activation error scores and the simulated naming latencies have

little effect when the differences are large (e.g. between regular and exception words), they become crucial when the differences are smaller (e.g. between consistent and inconsistent regular words).

For each case in Table 14, the differences are more significant for the Glushko word set than for the Taraban & McClelland word set. It thus appears that the differences between the Glushko and Taraban & McClelland experimental results may be put down to the particular word sets used. The model clearly predicts that a regular inconsistent effect should occur in humans. Again we would expect a frequency effect as well, but this has probably been much reduced by the logarithmic word frequency compression used by the model. The suggestion by Seidenberg & McClelland (1989) that Glushko's results were merely an artefact of inadvertent intralist priming effects now seems less likely.

Strange Words.

Another interesting class of words (termed *strange*) are those that contain very rare spelling patterns (e.g. 'aisle' → /Il/ and 'once' → /w^ns/). Waters & Seidenberg (1989) have compared the naming latencies of strange words with those of regular and exception words and found that strange words have an even larger type effect than exception words. Their results are summarized in Figure 28. For low frequency words the latencies are significantly longer for the strange words than for the exception words or regular controls. For the higher frequency words the differences are not significant.

There are essentially two reasons why we should find longer latencies for strange word reading: First, the text to phoneme correspondences are not only rare but also highly irregular so we could expect them to behave like particularly low frequency and particularly irregular words. Secondly, the low orthographic redundancy of the rare spelling patterns may result in slower individual letter recognition and hence a longer naming latency for that reason. Our model can be expected to exhibit the effects of the first reason but not of the second.

Figures 29 and 30 show the learning and error curves during training of our model for a set of strange words and the corresponding regular control words. As for the exception and regular inconsistent words, it is easy to see evidence of type and frequency effects. However, Figure 19 indicates that the size of the strange word effect we find in our model is somewhat less than that found in humans, in particular it is less than (rather than greater than) that for the exception words. It is possible to understand why: the strange words (because they are strange) experience less interference than the non-strange exception words do from the regular words and hence the learning can reduce the errors relatively unimpeded.

The model's full set of simulated naming latencies and statistical significances are shown in Table 15. We find highly significant type effects in all cases, but as for the exception and regular inconsistent words, most of the frequency effects fail to reach significance. Comparing the Waters & Seidenberg strange word results with those of the corresponding exception words (in Table 13) we again find that our choice of mathematical relation between the errors and simulated latencies is crucial. For the 'log error', 'inverse log error' and 'log Luce ratio' relations the strange word latencies are similar to, or below, the corresponding exception word latencies. For the 'sum inverse inputs' relation, the strange word latencies are greater than for the exception words as is found in the experiments.

Unique Words.

Brown (1987) has suggested that rather than simply distinguishing mono-syllabic words by how regular they are, it would be more revealing to consider how many friends and enemies their word bodies have. We then have three classes of words to compare: Consistent words that have many friends and no enemies (e.g. 'vent', 'bent', 'tent'), exception words that have no friends and many enemies (e.g. 'comb', cf. 'tomb', 'bomb') and *unique* words that have neither friends nor enemies (e.g. 'soap', 'bulb').

Figure 32 shows the naming latency results from Brown's experiment and the mean log error scores from our model. In the experiment there was found to be a significant difference between the exception and consistent words but no significant difference between the unique and exception words. The model's full set of simulated naming latencies and statistical significances of the differences are shown in Table 16. We find a consistent hierarchy of Exception > Unique > Consistent for all four sets of simulated naming latencies. This is precisely what we would expect from our consideration of the interference effects during training. All the word type differences are significant except for that between the unique and exception words for the 'sum inverse inputs' relation.

In a more comprehensive series of experiments Jared, McRae & Seidenberg (1990) showed that significant consistency effects *do* occur in humans, i.e. that the exception words *are* slowed relative to unique words. They argued that Brown failed to find these effects because his exception words were not exceptional enough, i.e. that the frequencies of the enemies of the exception words were not high enough relative to the actual exception word frequencies to cause sufficient inhibition. We have already seen that the word frequency compression used in our model's training data collapses all the frequency differences. This problem will have relatively little effect on the unique words but means that the exception word enemies in our model *will* be able to cause inhibition of the exception words.

In conclusion then, our model has effectively corrected for the problems in Brown's original experiment and is in general agreement with Jared et al.'s finding of a consistency effect in humans.

Non-Words.

Numerous experimental studies have found that naming latencies are longer for non-words than for real words (e.g. Forster & Chambers, 1973; Frederiksen & Kroll, 1976; Glushko, 1979). Figure 33 summarises the results of Glushko's experiments. The word versus non-word and regular versus exception differences were both found to be significant but there was no significant interaction effect.

Converting our network output error scores into simulated naming latencies is even more problematic for non-words than we have already found it to be for words in the training data. The first problem is that the network will not necessarily give a correct output for the non-words and so the output error scores can become very large. This should not be too much of a problem for the simple 'log error' or 'log sum Luce ratios' relations. However, the assumption of small errors that we used to justify our 'inverse log error' relation becomes invalid, so we will not be able to get sensible results for that. Also, when the output activations fall below 0.5 (as often happens for non-words), they correspond to negative inputs, so we cannot use the 'sum inverse inputs' relation at all.

To understand what is happening in our model for non-words we need to

extend slightly our previous discussion of interference effects during training. We know that the network has learnt the main GPC rules very well, so regular items, whether words or non-words, will have low output errors and hence low simulated naming latencies. Exception words will have been learnt correctly but interference from the regular words will result in increased output errors and hence longer simulated naming latencies. Non-words derived from exception words will have their outputs influenced by both the main GPC rules and by the relevant exception words. As already noted in our discussion of generalization, the resulting pronunciation will either be regular or analogous depending on the details of the rule hierarchy learnt by the network. Either way, the outputs will tend to be drawn between the regular and exception responses and will end up with higher error scores and hence higher simulated naming latencies than either.

The model's three remaining forms of simulated naming latencies for the Glushko experiment are shown in Table 17, along with the statistical significances of the various type differences. The mean log error scores for the model are shown in Figure 34. We find highly significant regular versus exception effects for both the words and non-words, which is easily understandable from our preceding discussions and consistent with the experiment. (Note particularly that it is very difficult to explain the regularity effect for non-words in terms of a traditional dual route model.) The word versus non-word differences are significant for the exceptions but not for the regulars and this is also apparent from the distributions shown in Figure 17. This is what we would expect from the above discussion but is in direct contradiction with the experiment.

It is difficult to see how any model with a good generalization performance could possibly produce naming latencies for regular non-words that are significantly different from those of regular words. It is clear that, like the related problem of lexical decision, the lexical/semantic route must have an important effect here and we cannot hope to account for the experimental findings until it has been incorporated into the model.

Pseudohomophones.

A particularly important class of non-words, termed *pseudohomophones*, are those which have the same pronunciation as a real word (e.g. 'kight', 'supe', 'trax'). McCann & Besner (1987) showed that such non-words have a significant pronunciation advantage over matched non-words which do not sound like real words. This 'pseudohomophone effect' is taken to indicate that some kind of lexicon must be consulted in the course of assembling a pronunciation. Since our model has no lexicon we should not expect it to exhibit a pseudohomophone effect.

To test this, the simulated naming latencies of our model were computed for the same sets of pseudohomophones, control non-words and control words that were used by McCann & Besner. The results and the statistical significances of the differences are shown in Table 18. As expected, we find a significant difference between the control non-words and control words and between the pseudo-homophones and control words but no significant difference between the pseudo-homophones and the control non-words. Once again we have evidence that reading models of this type require a lexical/semantic route in addition to their rule based route.

Conclusion.

It is evident that there are a number of fundamental difficulties in relating the

output activation error scores to simulated naming latencies in simple feedforward networks. Nevertheless, our model seems to be in broad agreement with many of the experimental naming latency results for humans. The remaining disagreements concerning regular non-words and pseudohomophones, however, suggest quite strongly that an additional lexical/semantic route must be added to our model in order to be in complete agreement with the experimental results. We shall discuss this point further in our concluding section.

If we compare our simulated naming latencies with those of the Seidenberg & McClelland (1989) model, it would appear that their model actually gives more realistic results than ours (particularly for the frequency effects) despite their much poorer generalization performance. However, it should be noted that the training of the Seidenberg & McClelland model was stopped when its performance on the training data was only 97.3%, at which point several of the words used in the naming latency test sets had still not been learnt correctly (Seidenberg & McClelland, 1989, Tables 1 & 2). This is equivalent to stopping the training of our network near the epoch 128 data point (where the training data performance is 97.6%). If we look at the mean error scores at this point on each of Figures 19, 21, 26 & 30, we see that the relations between the different word types and frequencies are remarkably similar to those of Seidenberg & McClelland and the experiments, with large type effects for the low frequency words and very little difference between the high frequency words and the two regular control sets. It is not clear at present whether this is an indication that the rule based route in humans gives up learning early and leaves the lexical/semantic route to take care of the difficult words, or simply that we should not place too much significance on the details of Seidenberg & McClelland's simulated naming latency results.

A final complication concerning our model's reduced frequency effect concerns our use of the over-training parameter *errcrit*, which may well be related to the precision with which real neural computation can be carried out. A non-zero value for *errcrit* introduces additional ceiling effects into the learning algorithm and so varying this will change the extent to which the various type and frequency effects will remain significant as the training progresses. Since the variations brought about by changing the word frequency representation and the value of *errcrit* we use in our model are so interrelated, a large number of simulations may be required to investigate these effects properly.

Learning and Developmental Dyslexia

In this section we consider the normal and abnormal reading development in our model and examine how well it corresponds to that of children.

Normal reading development in children is generally considered to proceed in stages (e.g. Frith, 1985). The first stage is dominated by a logographic strategy in which there is instant recognition of whole words. This results in comparable performance on regular and exception words and virtually no ability in reading unfamiliar words. In the second stage, an alphabetic strategy is developed in which words can be pronounced using an increasingly complex hierarchy of text to phoneme rules. The performance on regular words improves, as does the ability to read unfamiliar words, but there is a tendency to over-regularize exception words. Finally, essentially perfect performance is achieved on all word types. In terms of

the standard dual route model, these stages correspond to acquiring the lexical/ semantic route first, which is later dominated by the acquisition of the phonological route and finally some combination of both routes is used.

Developmental dyslexia is a term used to describe reading difficulties in children that are not otherwise disadvantaged (e.g. in terms of intelligence or opportunity). There is still considerable debate concerning the precise causes of such disorders and whether or not there is more to dyslexia than a simple slowness of development. In the developmental framework described above, classic developmental dyslexia is considered to correspond to a developmental "arrest" at the first stage (Frith, 1985). Such dyslexics have a much increased word versus non-word effect but no regularity effect, which is consistent with little or no development of the phonological route. Since our model *only* has a phonological route we cannot hope to account for such symptoms: If we look at all our learning curves (Figures 7, 10, 12, 13 and 15) we see that the ability to deal with non-words always develops at the same rate as the ability to read regular words and that it is only at a very late stage that the regularity effect becomes diminished. Although our model cannot account for the first stage, it seems to provide a reasonably good account of the second stage of normal reading development.

Another large class of poor readers (that are usually also classed as dyslexic) show a different set of symptoms. They exhibit a larger regularity effect than normal readers and are also able to produce acceptable pronunciations for many unfamiliar words and non-words (Backman et al., 1984; Castles & Coltheart, 1992). This would appear to indicate an "arrest" during the second stage of development, though it is not obvious whether such symptoms correspond to an actual disorder of the phonological route or merely a slowing of the passage through this stage. This is where our explicit model can be expected to throw some light.

It is dangerous to attempt to compare detailed performances because the effects in humans are usually obscured by a wide range of compensatory strategies which are not yet available to our connectionist models. We will therefore concentrate on exhibiting the possible mechanisms that may account for such symptoms in our model without attempting any detailed comparison with humans. In fact, we have already observed symptoms reminiscent of developmental dyslexia several times in the preceding discussions.

Seidenberg & McClelland (1989) suggested that the deterioration in performance caused by reducing the number of hidden units might provide an account for developmental dyslexia in their model. We have already investigated (in our discussion of the basic model) the dependence of our model's performance on the number of hidden units in the network. The resulting final performances for networks with very small numbers of hidden units (shown in Figure 35) indicate an inability to deal with exception words relative to regular words, similar to developmental dyslexics, in our model as well. Of course, it is not being suggested that developmental dyslexia is caused by the child having less than 30 active neurons, but rather that it could be caused by some general computational deficit equivalent to vastly reducing the number of neurons.

Perhaps the most drastic reduction in computational ability was considered when we removed the hidden units completely and had only direct input to output connections. It is well known that such networks can only handle linearly separable problems (Minsky & Papert, 1969) and Figure 36 shows that this affects the networks ability to handle exceptional words much more than regular words and non-words.

Again, we are not suggesting that dyslexic children might not have any hidden neurons, but merely demonstrating that an inability to form adequate internal representations necessary to deal with non-linearly separable problems will also result in the preferential inability to deal with exception words compared with regular words and non-words.

In our discussion of the basic model we noted how the standard back-propagation learning algorithm could get stuck and fail to acquire the sub-rules and exceptions once the main rules had been learnt. Figure 37 shows the resulting final performances which are again similar to those of our developmental dyslexics. Once again, we are not suggesting that real brains employ a Sigmoid Prime Offset but rather that their reading difficulties could equally well arise from particular aspects of the learning process as from general limitations on the computational resources.

Finally, we need to consider if it is possible to distinguish these three 'deviant' accounts of developmental dyslexia from a simple 'delayed' account. In Figure 38 we show the network's performance at three points during learning and again we find a large difference between the regular and exception words. This constitutes our fourth, delayed, account. Given that each of our four accounts are merely approximations to more realistic scenarios, and also that human dyslexics tend to make use of a range of compensatory strategies, it is premature to choose between them, but it is reassuring to see that our model admits so many possibilities.

In fact, to a certain extent, it is not at all surprising that we get such similar results for the deviant and delayed accounts. We can actually relate what is happening in both cases to the same well known problem of 'over-training' or 'over-fitting' in neural networks that have too many free parameters for the data they are trying to model (Baum & Haussler, 1989). There is a natural tendency with any gradient descent learning algorithm (such as back-propagation) for all the hidden units to start off behaving in the same way. They all begin by learning to model the main regularities in the training data (i.e. the main rules) and it is only after this has been achieved that some of them begin to account for the less regular features in the data (i.e. the sub-rules and exceptions). In many real world situations, the less regular features in the training data actually correspond to errors (or noise) and (as we have seen in our model) learning these tends to reduce the generalization performance. Consequently, there have been numerous procedures proposed to prevent this second stage of learning from taking place in artificial neural networks. The obvious approach is simply to stop the training early or, equivalently, to look at the network's performance at an early stage of training (e.g. Morgan & Bourlard, 1990; Weigend, Huberman & Rumelhart, 1990). A similar effect is obtained by reducing the number of free parameters. This can be achieved by directly restricting the number of connection weights or hidden units, or by various indirect techniques such as network pruning (e.g. Mozer & Smolensky, 1989; Karnin, 1990), weight decay (e.g. Hinton, 1987; Krogh & Hertz, 1992) or weight sharing (e.g. Nowlan & Hinton, 1992). Clearly, in our case, the lack of a Sigmoid Prime Offset has a similar effect. Looking at the network performance from this point of view, we see that it may prove very difficult to distinguish between the delayed and various deviant accounts of developmental dyslexia since they all correspond to the prevention or reduction of the same second stage of learning.

Another important consequence of this insight is that all the human like developmental regularity effects will be independent of the details of the network

architecture and representations. Virtually any system that has an input to output mapping learnt by a gradient descent algorithm will result in qualitatively similar performance in this respect. This includes the original Sejnowski & Rosenberg (1987) NETtalk, the Seidenberg & McClelland (1989) Wickelfeature based model, the newer Plaut, McClelland & Seidenberg (1992) models and all the corresponding spelling models.

Damage and Acquired Dyslexia

An important method of constraining cognitive models is to examine their performance after damage (e.g. Shallice, 1988; Coltheart et al., 1993). In this section we examine how well our model stands up to these constraints.

Of particular importance for models of reading are two forms of acquired dyslexia: Patients with phonological dyslexia exhibit a dissociation between word and non-word naming, for example, W.B. (Funnell, 1983) showed a complete failure to read non-words whilst maintaining around 90% success on words. Patients with surface dyslexia exhibit a dissociation between regular and exception word naming, for examples, M.P. (Bub, Cancelliere & Kertesz, 1985) had 90% success on low frequency regular words against 40% on low frequency exceptions and H.T.R. (Shallice, Warrington & McCarthy, 1983) managed 80% on regular words against 35% on very irregular words. Thus there appears to be a double dissociation between lexical and phonological (or equivalently, exception word and non-word) reading and, by the standard inference of cognitive neuropsychology, this is taken to imply modularity of function (e.g. Shallice, 1988). This is, of course, one of the main reasons why so many people believe in forms of the dual route model of reading.

Nevertheless, it is not totally obvious that we cannot get a double dissociation in our model. Some small scale studies (Wood, 1978; Sartori, 1988; Bullinaria & Chater, 1993) have indicated that it *is* possible to obtain double dissociations in distributed systems and Dunn & Kirsner (1988) have shown that it *is* "possible to posit single processes that mimic both single and double dissociation". Against this is evidence that the double dissociations found in the small scale studies are merely artefacts of the small size that will not scale up and that the counter-examples constructed by Dunn & Kirsner are too far removed from realistic systems to be of relevance to real brains (Shallice, 1988; Bullinaria & Chater, 1994). We will keep an open mind on the matter and examine how our model performs after a range of different types of damage.

The effects of damage on existing neural network models of reading have already been examined. Sejnowski & Rosenberg (1987) examined the effect of random perturbations of the weights in their original NETtalk model but did not separately analyse the effects on different subsets of the words. Patterson, Seidenberg & McClelland (1989) carried out an analysis of the effects of removing random subsets of the connections and hidden units in the Seidenberg & McClelland (1990) model and found symptoms similar to certain forms of acquired surface dyslexia but had "nothing to say about phonological dyslexia".

There are three basic forms of network (brain) damage we can consider: (1) Changing the weights (synaptic strengths), e.g. by scaling, reducing, clipping or adding noise, (2) Removing connections (axons and dendrites), and (3) Removing hidden units (neurons). By studying the effect of all these forms of damage (six

types in all) we will get a fair indication of the range of effects available. Of course, a much larger range of types of damage is possible (e.g. by treating the thresholds separately from the other weights, by treating the input and output connections differently, by interfering with the activation functions, etc.). The (relatively minor) effects of these variations and small scale artefacts are discussed in Bullinaria & Chater (1994). We will begin by looking at the specific effects of our six types of damage and then consider the more general implications of what we find. The results we shall present are all for the Taraban & McClelland (1987) regular and exception word sets (which each consist of 48 words) and the Glushko (1979) regular non-word set (of 43 non-words). In each case the degree of damage is increased from zero to a level where the network fails to produce any correct outputs at all. Patients with varying degrees of acquired dyslexia will correspond to appropriate intermediate stages.

First we consider the deterministic weight modifications. Figure 39 shows the effect of a global reduction of all the network weights by successive applications of factors of 0.9. (This is equivalent to flattening all the sigmoidal activation functions.) For each trained network, this process is totally deterministic in the sense that each application of the process always gives the same result. We invariably find that initially the network is fairly resilient to damage but eventually the exception word performance begins to fall off and later the regular word performance falls off as well. The delay between the loss of the exception words and the regular words can result in a fairly large dissociation between the two word types. Moreover, the magnitudes of the largest dissociations are similar to those found in surface dyslexic patients. For example, at the point where all the weights have been reduced by an overall factor of 0.4 we have an 86% performance on the regular words but only 36% on the exception words. Figure 40 shows that we get a similar effect if we globally reduce all the weights by successive reductions by constant amounts of 0.04, though the size of the dissociations tend to be slightly smaller in this case. A third deterministic process of network damage is provided by weight clipping, i.e. by the imposition of successively smaller maximum allowable weights. Figure 41 shows the effect of doing this. We find that there is a slight preferential loss of the exception words over the regular words but nowhere near to the degree found in real dyslexic patients.

The most obvious form of non-deterministic weight modification damage is to change each of the weights by random amounts. Figure 42 shows the effect of adding Gaussian noise to the weights in successive amounts with zero mean and standard deviation of 0.2. To assess the degree of variability between the different sets of random weight changes, the process was repeated ten times for each of the *errcrit* = 0.0 and 0.0001 networks. The largest dissociation for each case with a regular word performance better than 70% are shown in Figure 43. Again we find symptoms similar to surface dyslexia and the degree of variability indicates that such damage could account for the effects found in real dyslexics.

We now come to the more radical forms of damage, namely the removal network connections. If this form of damage is localized, it is equivalent to removing whole hidden units and, by implication, all connections to and from them. The removal of connections at random corresponds to a more distributed form of damage. We will consider this first. Our networks contain approximately 132000 connections so we damaged the network by removing successive sets of 6600 connections, i.e. 5% of the total number. Computationally we can implement this by

simply setting the appropriate weights to zero. Figure 44 shows a typical set of damage curves. To illustrate the degree of variability between different sets of removed connections, the process was repeated eleven times for each of the *errcrit* = 0.0 and 0.0001 networks. The largest dissociation for each case with a regular word performance better than 70% are shown in Figure 44. Again we find a preferential loss of the exception words to a degree compatible with real dyslexic patients.

Finally we consider the effect of removing hidden units from our networks. Computationally this can be simulated by simply setting the appropriate thresholds to infinity. Figure 46 shows the typical effect of removing successive random sets of ten hidden units from the network. Again we find a clear dissociation between the regular and exception words, though the magnitude of the effect is slightly smaller than for the other forms of damage. Figure 47 shows the degree of variability over 13 different sets of removed hidden units for each of the *errcrit* = 0.0 and 0.0001 networks. In order to assess the maximum effect size that is likely to result from this form of damage, an additional damage run was performed with the hidden units removed by hand rather than at random. This was done by testing the effect of removing each unit one at a time and then actually removing the units which had the most detrimental effect on the exceptions and the least effect on the regulars. After about seven steps of removing about ten units at a time we arrived at the dissociations denoted by 'damage run H' on Figure 47. No doubt even larger dissociations could be obtained by a more careful and systematic application of this approach. Of course, the chances of random damage resulting in these optimal dissociations are pretty negligible, so there is no real point in pursuing this any further. The important point is that the range of effects produced by random damage, together with the size of the effect that we now know is possible, leads us to be quite confident that this form of damage could lead to the effects found in real dyslexics.

Apart from the effects of network damage we can learn other things from the removal of hidden units. In particular, by removing each hidden unit in turn, one at a time, we can get an idea how distributed the internal representations are. In fact, for both our networks, we found that there was a least one hidden unit that, when removed, resulted in output errors. Thus, even 300 hidden units (which is at least ten times the number actually needed to learn the training data) is not enough to ensure that the network is fully distributed in the sense that no single neuron has a significant effect on its performance. Nevertheless, looking at damage curves such as Figure 46, we see that the networks are still remarkably resilient with performances that degrade gracefully rather than catastrophically.

We have considered six forms of network damage and five of them result in symptoms compatible with acquired surface dyslexia. The sixth also has more exception words lost than regular words but to a much lesser extent than is often found in humans. The symptoms are also very similar to those of developmental dyslexia that we discussed in the previous section. There we noted that the regular versus exception dissociation was an almost inevitable consequence of the gradient descent learning algorithm and relatively independent of the other details of the model. One can argue that the same is true for acquired dyslexia. Networks with a reasonably large number of hidden units tend be quite resilient to all forms of damage (as can be seen from the first few points on all our damage curves) and consequently the infliction of small amounts of damage will cause relatively minor perturbations to the output activations. It follows that the outputs which will be

first affected by damage will be those that already have the highest output activation error scores since these will already have activations nearer the central region of the sigmoids where the units of highest activation will cross over and result in the wrong phonemes being output. Our discussion of the learning interference effects and the naming latencies for our best model and the developmental accounts in the previous section have already shown that before damage the error scores are significantly larger for the exception words than for the regular words. Consequently, it should be no surprise to find that we always get the errors in the exception words occurring before those in the regular words and hence that we always end up with symptoms similar to surface dyslexia.

In some patients, e.g. M.P. (Bub et al., 1985) and K.T. (McCarthy & Warrington, 1986), there are also pronounced frequency effects with most of the errors occurring on the low frequency exception words and very few on the high frequency words. This, together with our discussion of the effect of frequency on the error scores (concerning naming latencies and interference effects during training), is further evidence that we have a correct understanding of what is happening. Unfortunately, as noted previously, the frequency effects in our model have been reduced considerably by the logarithmic word frequency compression in the training data (that was necessary for us to get the networks trained in a reasonable time) so most of the frequency effects in our damaged networks fail to reach significance.

With hindsight, then, it is almost obvious that we would end up modelling surface dyslexia when we damage our networks. It is almost as obvious that we cannot hope to model phonological dyslexia, since there is no way that our model can possibly output the correct phonemes for words without also doing so for similar non-words. This, together with our previous failure to account for lexical decision, the pseudohomophone effect and certain types of developmental dyslexia, suggests quite strongly that our model still needs to be accompanied by a separate lexical/semantic route which will bring us back towards a variation of the more traditional dual route models of reading (Coltheart et al., 1993). Phonological dyslexia will then simply correspond to the almost complete loss of our rule based route whilst the lexical/semantic route remains virtually intact.

Once we are forced to concede that our model is only the rule based route of a dual route model, there are some further subtleties that we need to consider. Shallice & McCarthy (1985) have classified patients that have a severely impaired ability to read aloud by means of the semantic system, into three broad categories in what is probably a continuum. First, for Type A patients such as W.L.P. (Schwartz, Saffran & Marin, 1980) we have:

1. Ability to read nearly all words,
2. Reading speed near normal.

Type B patients such as H.T.R. (Shallice, Warrington & McCarthy, 1983), M.P. (Bub, Cancelliere & Kertesz, 1985), K.T. (McCarthy & Warrington, 1986) and W.L.P. at later stages showed the following characteristics:

1. Near normal on regular words, poor on exception words,
2. Most errors are regularizations,
3. Near normal on non-words,
4. Reading speed near normal.

Finally, Type C patients which are more varied and include J.C. & S.T. (Marshall & Newcombe, 1973), exhibited the following characteristics:

1. Impaired reading of regular words, even worse on exception words,
2. Errors are not necessarily perfect regularization,
3. Reading is slow, often with corrections.

Understanding these three broad categories is now straightforward. Type A corresponds to a damaged semantic/lexical route but intact rule based route, Type B would correspond to a damaged semantic/lexical route with an intermediate degree of damage to the rule based route and Type C would correspond to severe damage of both routes. Table 19 summarizes the extent to which our network's exception word errors are regularizations for the six different types of damage. With the exception of the weight clipping case (which we have already determined does not give the required regularity effect anyway) we find similar regularization effects to real Type B patients. As we increase the damage the regularization errors get swamped by more serious errors as found for Type C patients.

Needless to say, real patients do not always have such clear cut symptoms as we have outlined above and often the data is confused by the patients' adoption of a range of compensatory procedures. For example, E.S.T. (Kay & Patterson, 1985) was faster and more accurate at reading non-words aloud than real words. However, if we also allow the possibility that the output from a slightly impaired semantic/lexical route may combine with that from our rule based route either to improve or impair the overall word reading performance, then virtually any combination of symptoms is possible.

Another possibility we need to consider is that, if our model has to be supplemented by an additional semantic/lexical route, then there is no real necessity for our model to learn to cope with all the exception words. The lexical/semantic route will have to learn the exception words anyway and overall efficiency might then mean that learning in the rule based route need not proceed beyond a certain stage (e.g. somewhere between the point where the generalization performance begins to level off and the point where the exception word performance becomes significant). Figure 38 shows the performance of our model near this point. If training does effectively stop here, then our model reduces to an implementation of the traditional phonological route of a dual route model with very poor exception word performance and both acquired and developmental surface dyslexia can be explained in the conventional dual route manner purely by loss or slow development of the semantic/lexical route.

Finally, it has often been noted that patients can recover remarkably quickly after suffering brain damage (e.g. Geshwind, 1985). The same is found to be true for our neural network models. On the removal of one set of 210 of its 300 hidden units our network's performance fell to 5.8% of the full training data set, 4.2% of the regular word set, 4.2% of the exception word set and 11.6% of the regular non-word set. After just one epoch of retraining, using only the remaining 90 hidden units, the performance had increased to 87.2% for the full word set, 97.9% for the regular words, 41.7% for the exception words and 97.7% for the non-words. It then took over 100 more epochs for the exception word ability to recover to its pre-damage levels. The interesting point is that the regular word versus exception word dissociation after a small amount of relearning following damage was actually a lot larger than at any point on the original damage curves.

Discussion and Conclusions

By training a simple feedforward neural network on a set of words with their corresponding pronunciations, we have arrived at a natural implementation of the rule based route of a dual route model of reading. Unlike in the *traditional* dual route model we are automatically able to achieve perfect performance on all words, including the irregular words, using the rule based route alone. However, because the network fails on the lexical decision task, cannot reproduce the pseudo-homophone effect and is unable to account for certain forms of developmental and acquired dyslexia, we are not able to consider it to be a realistic single route model on its own.

The model seems to be able to account for most aspects of human reading ability that we would normally expect of the rule based route of a dual route model. The network's generalization ability is comparable with human performance under time constrained conditions and, like humans, it pronounces certain non-words by analogy with exception words rather than by the main GPC rules. We find regularity effects during learning similar to humans and have identified several possible accounts of developmental dyslexia. When performing normally the networks' output activation error scores correlate well with numerous naming latency experiments on humans. Finally, as a result of several different forms of network damage we can understand various types of acquired surface dyslexia in humans. Moreover, it does all this without having to pre-process the training data by hand (which was necessary for the original NETtalk model of Sejnowski & Rosenberg, 1987) and without the use of complicated input-output representations such as Wickelfeatures (as used by Seidenberg & McClelland, 1989).

Despite this success the model is still deficient in several important respects. First, the model is still relatively small scale. We need to carry out further simulations with more hidden layers, more hidden units and larger training data sets. It is reasonable to expect that this will bring us the small improvements necessary to reach unconstrained human level generalization performance. It will also allow us to test the model more thoroughly on transcribed continuous speech containing multi-syllabic words and stress markers. We also have to see how well the network can cope with more informal speech that contains various errors and inconsistencies. Finally, we need to find a way of training the network with a more realistic word frequency distribution. This is necessary to confirm or refute our suggestion that this is the sole reason for our model failing to reproduce the full extent of the naming latency and dyslexia frequency effects found in humans. Carrying out all these improvements requires little more than finding additional computational resources.

The main fundamental aspect of the model that needs modifying is the use of the moving window which is psychologically implausible in many respects. There are at least two ways in which we may be able to improve the model in this respect without degrading the existing performance too much. The moving window approach is just one, particularly simple, way of representing the time delayed context effects in neural networks. In principle we can instead handle the necessary context information by a series of recurrent connections such as used by Jordan (1986) or Elman (1990). Thus a sensible next stage of development of our model might be to re-implement the moving window in terms of recurrent connections. This should also allow a more realistic study of priming and long range dependency

effects. There is also mounting evidence (e.g. Hinton & Shallice, 1991; Plaut & Shallice, 1991; Plaut & McClelland, 1993) that the related introduction of basins of attraction into the output activation space can also improve performance considerably. A less drastic alternative is simply to reformulate the network in terms of a static window, i.e. correlated input and output buffers. This, however, is more of a re-interpretation than a re-implementation and simply shifts the problem out of our network and into the operation of the buffers.

Of course, as far as a complete model of reading is concerned, the important next stage is to combine the current model with a lexical/semantic route. There is still considerable debate on the details of this route (e.g. Coltheart et al., 1993) and we shall not pre-judge these issues here. Some important (though small scale) work has already been done on mapping orthography to semantics to phonology (e.g. Dell, 1986; Hinton & Shallice, 1991; Plaut & Shallice, 1992). Whether we need a more direct lexical route as well is not yet clear (hence our continual use of the term 'lexical/semantic route'). Either way, much more work will be required to scale everything up and to determine the appropriate way to link together the two routes of a dual route model and the speech production system.

Adding a lexical/semantic route to our model will not, however, simply result in a traditional dual route model of the type outlined by Coltheart et al. (1993). Unlike previously proposed rule based routes that could only produce regular pronunciations, our neural network rule based route can correctly pronounce all word types and so there are no longer any conflicts between the outputs of the lexical/semantic route and the rule based route. This results in a simplification of the traditional dual route model in that we no longer have to postulate a separate sub-process to reconcile the conflicting outputs from the two routes for exception words. The two routes now each produce their share of activation into the phoneme output system which simply adds it up until the required threshold is reached. We have already seen that the rule based route alone can account for most of the regularity, consistency and frequency effects found in humans. (The extent to which there is also a contribution to the frequency and other effects from the semantic/lexical route will depend on the details of that route. There is no reason to suppose that they will conflict to any large degree with those from the rule based route.) There were only two major naming latency problems remaining, namely regular non-words and pseudohomophones. The difference between regular words and regular non-words is now easily explained by the fact that the real words have output contributions from two routes whereas the non-words get a contribution from the rule based route alone.

Explaining the pseudohomophone effect really requires a more complete model of the two routes and whatever 'speech production system' accepts their outputs. However, a few possibilities can be suggested at this stage. The basic effect is that non-words that sound like real words are pronounced faster than matched non-words which do not (McCann & Besner, 1987). First, it is possible that the speech production system is better at processing phoneme strings it knows (i.e. real words) than phoneme strings it does not (i.e. non-words). This training data versus generalization performance difference could arise naturally in virtually any neural network implementation. The same effect could also occur within the rule based route itself since, in a network with recurrent connections, the context information may be circulated to some extent in terms equivalent to output phoneme strings. (Note, however, that there is no evidence of this occurring in the recurrent networks

tested thus far.) A third, more circuitous possibility, is that the partially activated phoneme outputs from the rule based route could feed back into the semantic/lexical route which, by some process of pattern completion, could then itself contribute towards the phonemic outputs. There is already experimental evidence that the spelling to sound to meaning route plays a role in normal human reading (e.g. Van Orden, Johnston & Hale, 1988; Coltheart, Laxon, Rickard & Elton, 1988). The conflicting lexical decision evidence that this would produce is also consistent with the experimental finding that humans are slower to identify pseudohomophones as non-words than non-pseudohomophones (McCann, Besner & Davelaar, 1984). Note that all three of these possible pseudohomophone effect explanations are consistent with the rule based and semantic/lexical routes being totally modular.

Finally, we need to reconsider the way that our model deals with homographs. Originally we assumed that suitable context information would feed into our rule based system from 'somewhere else' and we showed that this was sufficient to resolve the ambiguities. Now that we have a separate semantic/lexical route that also provides phonemic activation, there are other possibilities. The simplest explanation is that the ambiguity results in the homophone outputs from the rule based route having similar low levels of activation for all the possible pronunciations leaving it for the semantic/lexical route to push the appropriate one across the threshold first. This is consistent with experimental evidence of particularly long naming latencies for isolated homographs (Seidenberg, Waters, Barnes & Tanenhaus, 1984). Again, we require no direct interaction between the two routes.

In conclusion then, it seems fair to say that a class of neural network models have been presented which, given their simplicity and room for improvement, appear to provide a promising basis for the rule based route of a realistic dual route model of reading.

Acknowledgements

This research was supported by the United Kingdom Joint Councils Initiative in Cognitive Science/HCI, grant number SPG 9029590. Various aspects of this work have benefited greatly from discussions with Nick Chater and other members of the Edinburgh University Connectionism and Cognition Research Group.

References

Andrews, S. (1982). Phonological recoding: Is the regularity effect consistent? *Memory & Cognition*, **10**, 565-575.

Backman, J., Bruck, M., Hébert, M. & Seidenberg, M. (1984). Acquisition and use of spelling-sound information in reading, *Journal of Experimental Child Psychology*, **38**, 114-133.

Baron, J. & Strawson, C. (1976). Use of Orthographic and Word-Specific Knowledge in Reading Words Aloud, *Journal of Experimental Psychology: Human Perception and Performance*, **2**, 386-393.

Baum, E.B. & Haussler, D. (1989). What Size Net Gives Valid Generalization? *Neural Computation*, **1**, 151-160.

Besner, D., Twilley, L., McCann, R.S. & Seergobin, K. (1990). On the Connection Between Connectionism and Data: Are a Few Words Necessary?, *Psychological Review*, **97**, 432-446.

Brown, G.D.A. (1987). Resolving Inconsistency: A Computational Model of Word Naming, *Journal of Memory and Language*, **26**, 1-23.

Bub, D., Cancelliere, A. & Kertesz, A. (1985). Whole-word and analytic translation of spelling to sound in a non-semantic reader, In K.E. Patterson, J.C. Marshall and M. Coltheart (eds.) *Surface dyslexia: neuropsychological and cognitive studies of phonological reading*, Erlbaum, London.

Bullinaria, J.A. (1993a). Neural Network Models of Reading Without Wickelfeatures, *Proceedings of the 2nd Neural Computation and Psychology Workshop*, Edinburgh.

Bullinaria, J.A. (1993b). Neural Network Learning from Ambiguous Training Data, submitted to *Connection Science*.

Bullinaria, J.A. & Chater, N. (1993). Double Dissociation in Artificial Neural Networks: Implications for Neuropsychology, *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, 283-288, Hillsdale, NJ: Erlbaum.

Bullinaria, J.A. & Chater, N. (1994). Connectionist Modelling: Implications for Cognitive Neuropsychology, submitted to *Language and Cognitive Processes*.

Castles, A. & Coltheart, M. (1992). Phonic knowledge, whole word knowledge and learning to read, *Literary Round Table Monograph*, Sydney: Board of Studies.

Cleeremans, A. & McClelland, J.L. (1991). Learning the structure of event sequences, *Journal of Experimental Psychology: General*, **120**, 235-253.

Cohen, J., Dunbar, K. & McClelland (1990). On the control of automatic processes: A parallel distributed processing model of the Stroop task, *Psychological Review*, **97**, 332-361.

Coltheart, M., Curtis, B. Atkins, P. & Haller, M. (1993). Models of Reading Aloud: Dual-Route and Parallel-Distributed-Processing Approaches, *Psychological Review*, **100**, 589-608.

Coltheart, V., Laxon, V., Rickard, M. & Elton, C. (1988). Phonological recoding in reading for meaning by adults and children, *Journal of Experimental Psychology Human Perception and Performance*, **14**, 387-397.

Dell, G.S. (1986). A Spreading-Activation Theory of Retrieval in Sentence Production, *Psychological Review*, **93**, 283-321.

Deprit, E. (1989). Implementing Recurrent Back-Propagation on the Connection Machine, *Neural Networks*, **2**, 295-314.

Dunn, J.C. & Kirsner, K. (1988). Discovering functionally independent mental processes: The principle of reversed association, *Psychological Review*, **95**, 91-101.

Elman, J.L. (1990). Finding Structure in Time, *Cognitive Science*, **14**, 179-211.
Fahlman, S. E. (1988). Faster-Learning Variations on Back-Propagation: An Empirical Study, in *Proceedings of the 1988 Connectionist Models Summer School,* Los Altos, CA: Morgan Kauffmann.
Forster, K.I. & Chambers, S. (1973). Lexical Access and Naming Time, *Journal of Verbal Learning and Verbal Behavior*, **12**, 627-635.
Frith, U. (1980). *Cognitive Processes in Spelling*, London: Academic Press.
Frith, U. (1985). Beneath the Surface of Developmental Dyslexia. In K.E. Patterson, J.C. Marshall and M. Coltheart (eds.) *Surface dyslexia: neuropsychological and cognitive studies of phonological reading* , London: Erlbaum.
Funnell, E. (1983). Phonological processes in reading: new evidence from acquired dyslexia, *British Journal of Psychology*, **74**, 159-180.
Frederiksen, J.R. & Kroll, J.F. (1976). Spelling and Sound: Approaches to the Internal Lexicon, *Journal of Experimental Psychology: Human Perception and Performance*, **2**, 361-379.
Geshwind, N. (1985). Mechanisms of change after brain lesions, *Annals of the New York Academy of Sciences*, **457**, 1-11.
Glushko, R.J. (1979). The Organization and Activation of Orthographic Knowledge in Reading Aloud, *Journal of Experimental Psychology Human Perception and Performance*, **5**, 674-691.
Henderson, L. (1982). *Orthography and word recognition in reading*. London: Academic Press.
Hinton, G.E. (1987). Learning Translation Invariant Recognition in a Massively Parallel Network. In G. Goos & J. Hartmanis (eds.) *PARLE: Paralell Architectures and Languages Europe. Lecture Notes in Computer Science*, pages 1-13. Berlin: Springer-Verlag.
Hinton, G.E. & Shallice, T. (1991). Lesioning an Attractor Network: Investigations of Acquired Dyslexia, *Psychological Review*, **98**, 74-95.
Humphreys, G.W. & Evett, L.J. (1985). Are there independent lexical and nonlexical routes in word processing? An evaluation of dual-route theory of reading, *The Behavioral and Brain Sciences*, **8**, 689-740.
Jared, D., McRae, K. & Seidenberg, M.S. (1990). The Basis of Consistency Effects in Word Naming, *Journal of Memory and Language*, **29**, 687-715.
Jordan, M.I. (1986). Attractor Dynamics and Parallelism in a Connectionist Sequential Machine, *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, 531-536, Hillsdale, NJ: Erlbaum.
Karnin, E.D. (1990). A Simple Procedure for Pruning Back-Propagation Trained Neural Networks. *I.E.E.E. Transactions on Neural Networks*, **1**, 239.
Kay, J. & Patterson, K. (1985). Routes to meaning in surface dyslexia. In K.E. Patterson, J.C. Marshall and M. Coltheart (eds.) *Surface dyslexia: neuro-psychological and cognitive studies of phonological reading* , London: Erlbaum.
Kreiner, D.S. & Gough, P.B. (1990). Two Ideas about Spelling: Rules and Word-Specific Memory, *Journal of Memory and Language*, **29**, 103-118.
Krogh, A. & Hertz, J. A. (1992). A Simple Weight Decay Can Improve Generalization. In J. E. Moody, S. J. Hanson & R. P. Lippman (eds.) *Advances in Neural Information Processing Systems 4*, pages 950-957. San Mateo, CA: Morgan Kauffmann.
Kucera, H., & Francis, W.N. (1967). *Computational analysis of present day American-English*,, Providence, RI: Brown University Press.

Lucasen, J.M. & Mercer, R.L. (1984). An information theoretic approach to the automatic determination of phonemic baseforms, *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 42.5.1-42.5.4.

Luce, R.D. (1963). Detection and Recognition. In R.D. Luce, R.R. Bush & E. Galanter (eds.), *Handbook of Mathematical Psychology* (Vol. 1), New York: Wiley.

Marshall, J.C. & Newcombe, F. (1973). Patterns of paralexia: A psycholinguistic approach, *Journal of Psycholinguistic Research*, **2**, 175-199.

McCann, R.S. & Besner, D. (1987). Reading Pseudohomophones: Implications for Models of Pronunciation Assembly and the Locus of Word-Frequency Effects in Naming, *Journal of Experimental Psychology: Human Perception and Performance*, **13**, 14-24**.**

McCann, R.S., Besner, D. & Davelaar, E. (1988). Word recognition and identification: Do word-frequency effects reflect lexical access? *Journal of Experimental Psychology: Human Perception and Performance*, **14**, 693-706**.**

McCarthy, R. and Warrington, E.K. (1986). Phonological reading: phenomena and paradoxes. *Cortex*, **22**, 359-380.

McClelland, J.L. (1979). On the time relations of mental processes: An examination of systems of processing in cascade. *Psychological Review*, **86**, 287-330.

McClelland, J.L. & Rumelhart, D.E. (1981). An interactive activation model of context effects in letter perception: Part 1. An account of basic findings. *Psychological Review*, **88**, 375-407.

Minsky, M. and Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*, Cambridge MA: M.I.T. Press.

Morgan, N. & Bourlard, H. (1990). Generalization and Parameter Estimation in Feedforward Nets: Some Experiments. In D.S. Touretzky (ed.) *Advances in Neural Information Processing Systems 2*, 630-637. San Mateo, CA: Morgan Kauffmann.

Mozer, M.C. & Smolensky, P. (1989). Using Relevance to Reduce Network Size Automatically. *Connection Science*, **1**, 3-16.

Norris, D. (1993). A quantitative model of reading aloud, Technical report, MRC APU, Cambridge.

Nowlan, S.N. & Hinton, G.E. (1992). Simplifying Neural Networks by Soft Weight Sharing. *Neural Computation*, **4**, 473-493.

Patterson, K., Seidenberg, M.S., & McClelland, J.L. (1989). Connections and disconnections: acquired dyslexia in a computational model of reading processes, in *Parallel Distributed Processing Implications for Psychology and Neurobiology.* (ed. R.G.M. Morris) Oxford: Oxford University Press.

Plaut, D.C. & McClelland,J.L. (1993). Generalization with Componential Attractors: Word and Nonword reading in an Attractor Network, *Proceedings of the 15th Annual Conference of the Cognitive Science Society*, 824-829, Hillsdale, NJ: Erlbaum.

Plaut, D.C., McClelland, J.L. & Seidenberg, M.S. (1992). Reading Exception Words and Pseudowords: Are Two Routes Really Necessary? Presented at the *Annual Meeting of the Psychonomic Society*, St. Louis, MO, November 1992.

Plaut, D.C. and Shallice, T. (1992). Deep Dyslexia: A Case Study of Connectionist Neuropsychology, submitted to *Journal of Cognitive Neuroscience.*

Rumelhart, D.E., Hinton, G.E. & Williams, R.J. (1986). Learning Internal Representations by Error Propagation. In D.E. Rumelhart & J.L. McClelland (eds.) *Parallel Distributed Processing*, Volume 2, Cambridge, Mass: MIT Press.

Rumelhart, D.E. & McClelland, J.L. (1986). On learning the past tenses of English verbs. In D.E. Rumelhart & J.L. McClelland (eds.)*Parallel Distributed Processing*, Volume 2, Cambridge, Mass: MIT Press.

Sartori, G. (1988). From Neuropsychological data to theory and vice versa. In G. Denes, P. Bisiacchi, C. Semenza, E. Andrewsky (eds.), *Perspectives in cognitive neuropsychology*, London: Erlbaum.

Schwartz, M.F., Saffran, E.M. & Marin, O.S.M. (1980). Fractionating the reading process in dementia: Evidence for word specific print-to-sound associations. In M. Coltheart, K.E. Patterson & J.C. Marshall (eds.), *Deep Dyslexia*, London: Routledge & Kegan Paul.

Seidenberg, M.S. & McClelland, J.L. (1989). A distributed, developmental model of word recognition and naming, *Psychological Review*, **96**, 523-568.

Seidenberg, M.S. & McClelland, J.L. (1990). More words but still no lexicon: Reply to Besner et al. (1990), *Psychological Review*, **97**, 447-452.

Seidenberg, M.S., Waters, G.S., Barnes, M.A. & Tanenhaus, M.K. (1984). When does irregular spelling or pronunciation influence word recognition? *Journal of Verbal Learning and Verbal Behavior*, **23**, 383-404.

Sejnowski, T.J. & Rosenberg, C.R. (1987). Parallel Networks that Learn to Pronounce English Text, *Complex Systems*, **1**, 145-168.

Shallice, T. (1988). *From neuropsychology to mental structure*. Cambridge: Cambridge University Press.

Shallice, T. & McCarthy, R. (1985). Phonological Reading: From Patterns of Impairment to Possible Procedures. In K.E. Patterson, J.C. Marshall and M. Coltheart (eds.) *Surface dyslexia: neuropsychological and cognitive studies of phonological reading* , London: Erlbaum.

Shallice, T., Warrington, E.K. & McCarthy, R. (1983). Reading without semantics, *Quarterly Journal of Experimental Psychology*, **35A**, 111-138.

Sullivan, K.P.H. & Damper, R.I. (1992). Novel-Word Pronunciation within a Text-to-Speech System, In G. Baily, C. Benoît & T.R. Sawallis (eds.),*Talking Machines: Theories, Models and Designs*, Amsterdam: Elsevier.

Svartvik, J. & Quirk, R. (1980). *A Corpus of Engish Conversation*, Lund: Gleerup.

Taraban, R. & McClelland, J.L. (1987). Conspiracy Effects in Word Pronunciation, *Journal of Memory and Language*, **26**, 608-631.

Treiman, R. & Zukowski, A. (1988). Units in Reading and Spelling, *Journal of Memory and Language*, **27**, 466-477.

Van Orden, G.C., Johnston, J.C. & Hale, B.L. (1988). Word identification in reading proceeds from spelling to sound to meaning, *Journal of Experimental Psychology Human Perception and Performance*, **14**, 371-386.

Waters, G.S. & Seidenberg, M.S. (1985). Spelling-sound effects in reading: Time-course and decision criteria, *Memory & Cognition*, **13**, 557-572.

Weigend, A.S., Huberman, B.A. & Rumelhart, D.E. (1990). Predicting the Future: A Connectionist Approach. *International Journal of Neural Systems*, **1**, 193.

Wickelgren, W.A., (1969). Context-sensitive coding, associative memory and serial order in (speech) behavior, *Psychological Review*, **76**, 1-15.

Wood, C.C. (1978). Variations on a theme of Lashley: Lesion experiments on the neural model of Anderson, Silverstein, Ritz & Jones, *Psychological Review,* **85,** 582-591.

Non-word	*Wrong Outputs*	*Frequency*	*Likely reason*
wosh	w*S	7	confusion with ‘wash’
	wOS	1	
	wUS	1	
mune	myn	3	problem with u -> yU
	mynl	3	
	myUl	1	
	mynn	1	
mone	mw^n	3	problematic analogy
	mw^	2	with 'one'
	mwnn	2	
wone	wOnn	3	problematic analogy
	ww^	1	with 'one'
	wO^n	1	
	w^nn	1	
	wwnn	1	
zute	zyt	4	problem with u -> yU
	myt	1	
	zUk	1	
tolph	tOl	1	too many sub-rules
	tlf	1	at once?
	t^lS	1	
	t^lf	1	
	tolf	1	
	tOlff	1	
fues	fyez	3	problem with u -> yU
	fyz	2	
tuel	tUUl	3	problem with u -> yU
	tUel	1	

Table 1. *The most common non-word pronunciation errors from nine successful runs and the likely reasons for those errors.*

	Regular	Exception	Control	Total
Non-recurrent				
40	93.0	95.3	85.0	89.8
80	95.3	90.7	95.0	94.0
120	93.0	93.0	93.7	93.4
Average	93.8	93.0	91.2	92.4
Recurrent				
40	95.3	97.7	90.0	93.4
80	93.0	97.7	95.0	95.2
120	93.0	97.7	91.2	93.4
Average	93.8	97.7	92.1	94.0

Table 2. *Comparison of non-word performances by matched recurrent and non-recurrent networks with 40, 80 and 120 hidden units.*

errcrit = 0.0001

	Regular	Exception	Control	Total
Correct	97.7	100.0	95.0	97.0
Regular	97.7	62.8	91.2	85.5
Competitor	0.0	27.9	10.0	12.0

errcrit = 0.0

	Regular	Exception	Control	Total
Correct	97.7	100.0	98.7	98.8
Regular	97.7	65.1	93.7	87.3
Competitor	4.7	11.6	10.0	9.0

Average

	Regular	Exception	Control	Total
Correct	97.7	100.0	96.9	97.9
Regular	97.7	64.0	92.5	86.4
Competitor	2.3	19.8	10.0	10.5

Overlap

	Regular	Exception	Control	Total
Correct	97.7	100.0	95.0	96.4
Regular	97.7	55.8	88.7	82.5
Competitor	0.0	7.0	6.2	4.8

Humans

	Regular	Exception	Control	Total
Correct	93.8	95.9		
Regular	93.8	78.3	88.6	87.3

Table 3. *The percentage performances on non-words for our two best neural networks and for humans.*

errcrit = 0.0001

	Regular	Exception	Control	Total
Correct	97.7	100.0	97.5	98.2
Regular	97.7	81.4	95.0	92.1

errcrit = 0.0

	Regular	Exception	Control	Total
Correct	97.7	97.7	96.2	97.0
Regular	97.7	83.7	91.3	91.0

Average

	Regular	Exception	Control	Total
Correct	97.7	98.8	96.9	97.9
Regular	97.7	82.6	93.1	91.6

Humans

	Regular	Exception	Control	Total
Correct	93.8	95.9		
Regular	93.8	78.3	88.6	87.3

Table 4. *The percentage performances on non-words for our two best neural networks after only 32 epochs of training (which is near the point of best generalization performance).*

a		__	0.716781
b	b	b_	0.000004
c	k	k_	0.000010
d	d	d_	0.000000
e		__	0.000001
f	f	f_	0.000000
g	g	g_	0.000104
h		__	0.000000
i		__	0.000119
j	dZ	dZ	0.000000
k	k	k_	0.000000
l	l	l_	0.000000
m	m	m_	0.000000
n	n	n_	0.000000
o	0	0_	0.612665
p	p	p_	0.000000
q	k	k_	0.037212
r	r	r_	0.000000
s	s	s_	0.000000
t	t	t_	0.000000
u		__	0.000245
v	v	v_	0.001704
w		__	0.000000
x	ks	ks	0.003740
y		__	0.008135
z	z	z_	0.021952

Table 5. *The network's default outputs for individual letters and the corresponding output activation error scores.*

|CaC|	•a•	__•_a_•___	0.000000
|CeC|	•e•	__•_e_•___	0.000000
|CiC|	•i•	__•_i_•___	0.000000
|CoC|	•o•	__•_o_•___	0.000000
|CuC|	•^•	__•_^_•___	0.000000
|CyC|	•i•	__•_i_•___	0.000000
|CaCe|	•A•	__•_A_•_____	0.000000
|CeCe|	•E•	__•_E_•_____	0.000731
|CiCe|	•I•	__•_I_•_____	0.000000
|CoCe|	•O•	__•_O_•_____	0.000000
|CuCe|	•yU•	__•yU_•_____	0.187645
|CyCe|	•I•	__•_I_•_____	0.000000
|CaCCe|	•a•	__•_a___•_____	0.006153
|CeCCe|	•e••	__•_e_•_•_____	0.000000
|CiCCe|	•E••	__•_E_•_•_____	0.436654
|CoCCe|	•o•	__•_o___•_____	0.000016
|CuCCe|	•^••	__•_^_•_•_____	0.000764
|CyCCe|	•y•s	__•_y_•_s_____	0.707352

Table 6. *The vowel sounds in different contexts provided by the generic consonant 'C' and word end marker '\|'. The symbol '•' indicates that no output activation was greater than 0.1. The errors shown are those for the main vowel.*

ai	A	A___	0.000000
au	*	*___	0.290617
aw	*	*___	0.000000
ay	A	A___	0.000000
ea	E	E___	0.337276
ee	E	E___	0.000000
ei	A	A___	0.310957
eu		____	0.000972
ew	U	U___	0.720870
ey		____	0.000005
ia	•	•___	0.921550
ie		____	0.164641
oa	O	O___	0.000000
oe	O	O___	0.000000
oi	Y	Y___	0.001049
oo	U	U___	0.010405
ou	W	W___	0.000009
ow	O	O___	0.111659
oy	Y	Y___	0.405144
ue		____	0.024988
ui		____	0.194051
uy		____	0.000000

Table 7. *The outputs produced by the double vowels.*

|CaiC|	•A	__•_A_____	0.000000
|CauC|	•*	__•_*_____	0.000000
|CawC|	•*	__•_*_____	0.000000
|CayC|	•A•	__•_A___•___	0.000000
|CeaC|	•E•	__•_E___•___	0.000000
|CeeC|	•E•	__•_E___•___	0.000000
|CeiC|	•E	__•_E_____	0.048072
|CeuC|	•U•	__•_U___•___	0.000121
|CewC|	•U•	__•_U___•___	0.000000
|CeyC|	•E•	__•_E___•___	0.000788
|CiaC|	•E	__•_E_____	0.023315
|CieC|	•E•	__•_E___•___	0.000011
|CoaC|	•O•	__•_O___•___	0.000000
|CoeC|	•O•	__•_O___•___	0.000000
|CoiC|	•Y	__•_Y_____	0.000010
|CooC|	•U•	__•_U___•___	0.000000
|CouC|	•W	__•_W_____	0.000001
|CowC|	•W	__•_W_____	0.000014
|CoyC|	•Y•	__•_Y___•___	0.016019
|CueC|	•U	__•_U_____	0.001308
|CuiC|	•U	__•_U_____	0.000006
|CuyC|	•U•	__•_U___•___	0.719078
|CaiCe|	•A•	__•_A___•_____	0.000000
|CauCe|	•*•	__•_*___•_____	0.178426
|CawCe|	•*•	__•_*___•_____	0.005579
|CayCe|	•Az	__•_A___z_____	0.000000
|CeaCe|	•E•	__•_E___•_____	0.000000
|CeeCe|	•E•	__•_E___•_____	0.000000
|CeiCe|	•E•	__•_E___•_____	0.084257
|CeuCe|	•U•	__•_U___•_____	0.121183
|CewCe|	•U•	__•_U___•_____	0.049362
|CeyCe|	•e•	__•_e___•_____	0.000468
|CiaCe|	•EA•	__•_E_A_•_____	0.000128
|CieCe|	•E•	__•_E___•_____	0.000003
|CoaCe|	•O•	__•_O___•_____	0.000000
|CoeCe|	•O•	__•_O___•_____	0.000000
|CoiCe|	•Y•	__•_Y___•_____	0.000002
|CooCe|	•U•	__•_U___•_____	0.000000
|CouCe|	•W•	__•_W___•_____	0.000277
|CowCe|	•O•	__•_O___•_____	0.048315
|CoyCe|	•Y•	__•_Y___•_____	0.002022
|CueCe|	•UE•	__•_U_E_•_____	0.558707
|CuiCe|	•UI•	__•_U_I_•_____	0.555042
|CuyCe|	•I•	__•___I_•_____	0.007322

Table 8. *The outputs produced by double vowels sandwiched between generic consonants 'C' and word end markers '\|'. The error scores are those for the double vowels.*

ca	ka	k_a_	0.000000
ce	s	s___	0.000000
ci	s	s___	0.026705
co	k	k___	0.000000
cu	k^	k_^_	0.000000
ch	S	S___	0.587988
ck	k	k___	0.000421
ght	t	____t_	0.000000
ng	N	N___	0.157011
ph	f	f___	0.510399
qu	kw	k_w_	0.003288
sh	S	S___	0.000000
th	T	T___	0.000006
lh	h	__h_	0.000000
lw	w	__w_	0.000000
ly	y	__y_	0.000050
lch	C	__C___	0.000025
lkn	n	____n_	0.000485

Table 9. *Over powering the main consonant GPC rules. The errors for the c-vowel combinations are for the 'c' sound only.*

|Cook|	•uk	__•_u___k___	0.000001
|CooC|	•U•	__•_U___•___	0.000000
|Cooke|	•Uk	__•_U___k_____	0.745265
|CooCe|	•U•	__•_U___•_____	0.000000
|CurC|	•er•	__•_e_r_•___	0.000000
|CurCe|	•er•	__•_e_r_•_____	0.000000
|CorC|	•Or	__•_O_r_____	0.000000
|CorCe|	•Or•	__•_O_r_•_____	0.000000
|ColC|	•Ol•	__•_O_l_•___	0.000001
|ColCe|	•ol•	__•_o_l_•_____	0.000410
|Coth|	•*T	__•_*_T_____	0.079397

Table 10. *Over powering the main vowel GPC rules. The errors are those for the main vowels.*

|pint|	pInt	__p_I_n_t___	0.001230
|pinC|	pin	__p_i_n_____	0.111940
|piCt|	pi t	__p_i___t___	0.027702
|Cint|	•int	__•_i_n_t___	0.000218
|piCC|	pi	__p_i_______	0.000027
|CiCt|	•i•t	__•_i_•_t___	0.000001
|CinC|	•in	__•_i_n_____	0.000001
|CiCC|	•i•	__•_i_•_____	0.000000
|mint|	mint	__m_i_n_t___	0.000021
|give|	giv	__g_i_v_____	0.000060
|giCe|	gI•	__g_I_•_____	0.055663
|Cive|	•Iv	__•_I_v_____	0.000007
|CiCe|	•I•	__•_I_•_____	0.000000
|dive|	dIv	__d_I_v_____	0.000028

Table 11. *Building up pronunciations for exception words. The errors are those for the main vowels.*

	short vowel		long vowel		
	generic	set	data	generic	set
ax	100.0	100.0	97.6	100.0	76.2
eb,ef,ek,ex	100.0	100.0	56.3	100.0	62.8
el	100.0	93.8	56.3	50.0	20.0
ix	100.0	100.0	99.2	100.0	75.0
oc,ox	100.0	100.0	89.9	100.0	92.5
uf,ux	100.0	100.0	91.4	100.0	98.8

Table 12. *The networks' ability to deal with the 'final e' rule in terms of percentages correct. 'Generic' refers to the generic constant C, 'set' refers to the full set of 40 different initial consonant clusters and 'data' indicates the percentage of the training data that follows the 'e rule' for that vowel.*

		Taraban & McClelland (4 × 24 words)	Waters & Seidenberg (4 × 10 words)
LOG ERROR	HF E	score = 6.45	score = 6.61
	HF C	score = 4.13	score = 4.89
	LF E	score = 7.09	score = 7.36
	LF C	score = 4.35	score = 5.01
	type(HF)	t = 10.11 p < 0.00001	t = 3.86 p = 0.00114
	type(LF)	t = 8.98 p < 0.00001	t = 4.62 p = 0.00021
	freq(E)	t = 3.83 p = 0.00037	t = 3.12 p = 0.00590
	freq(C)	t = 0.63 p = 0.52562	t = 0.19 p = 0.84987
	freq	F = 5.08 p = 0.02646	F = 1.67 p = 0.20398
	type	F =175.90 p < 0.00001	F = 36.25 p = 0.00001
	intn	F = 1.22 p = 0.27178	F = 0.87 p = 0.35599
−1 / LOG ERROR	HF E	score = 5.20	score = 5.42
	HF C	score = 3.15	score = 3.82
	LF E	score = 6.51	score = 7.20
	LF C	score = 3.39	score = 3.97
	type(HF)	t = 9.49 p < 0.00001	t = 3.18 p = 0.00522
	type(LF)	t = 7.81 p < 0.00001	t = 4.09 p = 0.00068
	freq(E)	t = 3.28 p = 0.00195	t = 2.55 p = 0.02018
	freq(C)	t = 1.07 p = 0.28808	t = 0.24 p = 0.81470
	freq	F = 11.55 p = 0.00099	F = 4.25 p = 0.04651
	type	F =129.48 p < 0.00001	F = 26.63 p < 0.00001
	intn	F = 5.62 p = 0.01985	F = 3.04 p = 0.08943
SUM INVERSE LATENCIES	HF E	score = 5.78	score = 6.86
	HF C	score = 4.52	score = 4.79
	LF E	score = 6.38	score = 6.21
	LF C	score = 4.86	score = 5.04
	type(HF)	t = 2.75 p = 0.00839	t = 2.93 p = 0.00886
	type(LF)	t = 4.67 p = 0.00005	t = 2.13 p = 0.04721
	freq(E)	t = 1.50 p = 0.14123	t = -1.10 p = 0.28625
	freq(C)	t = 0.82 p = 0.41699	t = 0.37 p = 0.71506
	freq	F = 2.67 p = 0.10581	F = 0.21 p = 0.64654
	type	F = 23.81 p < 0.00001	F = 13.13 p = 0.00089
	intn	F = 0.82 p = 0.41699	F = 1.02 p = 0.31809
LOG SUM LUCE RATIO	HF E	score = 8.08	score = 8.18
	HF C	score = 7.07	score = 7.30
	LF E	score = 8.40	score = 8.54
	LF C	score = 7.21	score = 7.42
	type(HF)	t = 7.63 p < 0.00001	t = 3.73 p = 0.00152
	type(LF)	t = 8.71 p < 0.00001	t = 5.46 p = 0.00003
	freq(E)	t = 3.37 p = 0.00154	t = 2.52 p = 0.02148
	freq(C)	t = 0.89 p = 0.37905	t = 0.44 p = 0.66679
	freq	F = 5.91 p = 0.01700	F = 2.45 p = 0.12646
	type	F =133.69 p < 0.00001	F = 41.08 p < 0.00001
	intn	F = 0.78 p = 0.37870	F = 0.63 p = 0.43435

Table 13. *Simulated naming latencies for the exception words and their controls. For each word set and mathematical relation is shown the mean scores and the statistical significances.*

		Taraban & McClelland (4 × 24 words)	Glushko (4 × 20 words)
LOG ERROR	HF RI	score = 5.26	score = 5.46
	HF C	score = 4.57	score = 4.05
	LF RI	score = 5.67	score = 6.29
	LF C	score = 4.39	score = 3.76
	type(HF)	t = 2.08 p = 0.04271	t = 5.17 p < 0.00001
	type(LF)	t = 3.20 p = 0.00247	t = 6.76 p < 0.00001
	freq(RI)	t = 0.99 p = 0.32521	t = 2.29 p = 0.02763
	freq(RI)	t = -0.56 p = 0.57871	t = -1.00 p = 0.32150
	freq	F = 0.18 p = 0.67136	F = 1.31 p = 0.25580
	type	F = 14.40 p = 0.00026	F = 72.40 p < 0.00001
	intn	F = 1.27 p = 0.26326	F = 5.81 p = 0.01831
– 1 / LOG ERROR	HF RI	score = 4.06	score = 4.10
	HF C	score = 3.46	score = 3.10
	LF RI	score = 4.81	score = 5.46
	LF C	score = 3.35	score = 2.95
	type(HF)	t = 2.20 p = 0.03320	t = 4.72 p = 0.00003
	type(LF)	t = 3.29 p = 0.00192	t = 5.68 p < 0.00001
	freq(RI)	t = 1.59 p = 0.11863	t = 2.95 p = 0.00540
	freq(C)	t = -0.47 p = 0.64196	t = -0.95 p = 0.34825
	freq	F = 1.57 p = 0.21286	F = 6.12 p = 0.01558
	type	F = 15.63 p = 0.00015	F = 51.39 p < 0.00001
	intn	F = 2.69 p = 0.10411	F = 9.60 p = 0.00272
SUM INVERSE LATENCIES	HF RI	score = 4.91	score = 4.83
	HF C	score = 4.63	score = 4.03
	LF RI	score = 5.49	score = 5.76
	LF C	score = 5.12	score = 4.76
	type(HF)	t = 0.66 p = 0.51279	t = 2.27 p = 0.02891
	type(LF)	t = 0.88 p = 0.38458	t = 2.65 p = 0.01164
	freq(RI)	t = 1.36 p = 0.17926	t = 2.30 p = 0.02642
	freq(C)	t = 1.19 p = 0.24066	t = 2.23 p = 0.03104
	freq	F = 3.27 p = 0.07392	F = 10.27 p = 0.00198
	type	F = 1.19 p = 0.27861	F = 12.16 p = 0.00081
	intn	F = 0.03 p = 0.86153	F = 0.14 p = 0.69967
LOG SUM LUCE RATIO	HF RI	score = 7.56	score = 7.64
	HF C	score = 7.25	score = 7.12
	LF RI	score = 7.84	score = 8.09
	LF C	score = 7.06	score = 6.96
	type(HF)	t = 1.79 p = 0.07938	t = 3.91 p = 0.00036
	type(LF)	t = 4.09 p = 0.00017	t = 6.10 p < 0.00001
	freq(RI)	t = 1.45 p = 0.15411	t = 2.40 p = 0.02117
	freq(C)	t = -1.07 p = 0.29210	t = -1.25 p = 0.22037
	freq	F = 0.14 p = 0.70927	F = 1.50 p = 0.22389
	type	F = 17.82 p = 0.00006	F = 52.40 p < 0.00001
	intn	F = 3.21 p = 0.07658	F = 7.18 p = 0.00902

Table 14. *Simulated naming latencies for the regular inconsistent words and their controls. For each word set and mathematical relation is shown the mean scores and the statistical significances.*

		Seidenberg & McClelland (4 × 23words)	Waters & Seidenberg (4 × 11 words)
LOG ERROR	HF S	score = 5.27	score = 5.77
	HF C	score = 4.22	score = 4.75
	LF S	score = 5.92	score = 7.23
	LF C	score = 4.64	score = 4.84
	type(HF)	t = 2.96 p = 0.00488	t = 1.95 p = 0.06490
	type(LF)	t = 4.25 p = 0.00011	t = 4.57 p = 0.00018
	freq(S)	t = 1.97 p = 0.05476	t = 3.52 p = 0.00215
	freq(C)	t = 1.25 p = 0.21716	t = 0.13 p = 0.89080
	freq	F = 5.17 p = 0.02540	F = 4.35 p = 0.04337
	type	F = 25.08 p < 0.00001	F = 21.31 p = 0.00004
	intn	F = 0.23 p = 0.63182	F = 3.44 p = 0.07085
– 1 / LOG ERROR	HF S	score = 4.05	score = 4.45
	HF C	score = 3.23	score = 3.72
	LF S	score = 4.65	score = 3.98
	LF C	score = 3.51	score = 3.84
	type(HF)	t = 3.28 p = 0.00202	t = 1.55 p = 0.13638
	type(LF)	t = 3.85 p = 0.00037	t = 4.33 p = 0.00032
	freq(S)	t = 1.88 p = 0.06661	t = 3.98 p = 0.00073
	freq(C)	t = 1.25 p = 0.21717	t = 0.21 p = 0.83310
	freq	F = 5.10 p = 0.02635	F = 9.58 p = 0.00359
	type	F = 25.64 p < 0.00001	F = 20.11 p = 0.00006
	intn	F = 0.69 p = 0.40791	F = 7.88 p = 0.00767
SUM INVERSE LATENCIES	HF S	score = 6.33	score = 6.63
	HF C	score = 4.94	score = 4.72
	LF S	score = 6.72	score = 8.28
	LF C	score = 5.21	score = 4.86
	type(HF)	t = 2.43 p = 0.01904	t = 3.01 p = 0.00698
	type(LF)	t = 3.04 p = 0.00402	t = 3.57 p = 0.00193
	freq(S)	t = 0.65 p = 0.51613	t = 1.72 p = 0.10152
	freq(C)	t = 0.58 p = 0.56724	t = 0.20 p = 0.84026
	freq	F = 0.75 p = 0.38617	F = 2.40 p = 0.12939
	type	F = 14.66 p = 0.00024	F = 21.46 p = 0.00004
	intn	F = 0.02 p = 0.87476	F = 1.76 p = 0.19270
LOG SUM LUCE RATIO	HF S	score = 7.64	score = 7.93
	HF C	score = 7.08	score = 7.26
	LF S	score = 7.91	score = 8.59
	LF C	score = 7.30	score = 7.34
	type(HF)	t = 2.85 p = 0.00664	t = 2.69 p = 0.01389
	type(LF)	t = 4.57 p = 0.00004	t = 5.97 p < 0.00001
	freq(S)	t = 1.74 p = 0.08881	t = 3.50 p = 0.00226
	freq(C)	t = 1.67 p = 0.24950	t = 0.31 p = 0.75577
	freq	F = 4.09 p = 0.04606	F = 5.30 p = 0.02657
	type	F = 24.48 p < 0.00001	F = 34.94 p < 0.00001
	intn	F = 0.07 p = 0.79185	F = 3.21 p = 0.08079

Table 15. *Simulated naming latencies for the strange words and their controls. For each word set and mathematical relation is shown the mean scores and the statistical significances.*

		Brown (3 × 21 words)
LOG ERROR	Unique	score = 5.57
	Exception	score = 6.52
	Consistent	score = 4.26
	Exc/Unq	t = 2.51 p = 0.01625
	Unq/Con	t = 6.52 p = 0.00083
	Exc/Con	t = 4.26 p < 0.00001
−1 / LOG ERROR	Unique	score = 4.39
	Exception	score = 5.63
	Consistent	score = 3.27
	Exc/Unq	t = 2.87 p = 0.00650
	Unq/Con	t = 3.27 p = 0.00224
	Exc/Con	t = 6.84 p < 0.00001
SUM INWARD STRENGTHS	Unique	score = 5.63
	Exception	score = 6.12
	Consistent	score = 4.77
	Exc/Unq	t = 1.26 p = 0.21583
	Unq/Con	t = 2.08 p = 0.04418
	Exc/Con	t = 3.56 p = 0.00098
LOG SUM LUCE RATIO	Unique	score = 7.74
	Exception	score = 8.19
	Consistent	score = 6.95
	Exc/Unq	t = 2.40 p = 0.02110
	Unq/Con	t = 4.37 p = 0.00009
	Exc/Con	t = 7.32 p < 0.00001

Table 16. *Simulated naming latencies for Brown's Unique, Exception and Consistent words. For each mathematical relation is shown the mean scores and the statistical significances of the differences.*

		Glushko (4 × 43 words)
LOG ERROR	Reg NW	score = 4.90
	Reg W	score = 4.81
	Exc NW	score = 8.11
	Exc W	score = 6.71
	NW/W (Reg)	t = 0.24 p = 0.80850
	NW/W (Exc)	t = 4.07 p = 0.00011
	Reg/Exc (NW)	t = 6.84 p < 0.00001
	Reg/Exc (W)	t = 7.72 p < 0.00001
	NW/W	F = 93.04 p < 0.00001
	Reg/Exc	F = 8.05 p = 0.00511
	intn	F = 6.09 p = 0.01456
− 1 / LOG ERROR	Reg NW	score = 6.70
	Reg W	score = 3.76
	Exc NW	score =33.08
	Exc W	score = 5.85
	NW/W (Reg)	t = 1.71 p = 0.09143
	NW/W (Exc)	t = 3.79 p = 0.00028
	Reg/Exc (NW)	t = 3.58 p = 0.00578
	Reg/Exc (W)	t = 6.44 p < 0.00001
	NW/W	F = 14.88 p = 0.00016
	Reg/Exc	F = 16.73 p = 0.00007
	intn	F = 10.83 p = 0.00121
LOG SIMULATED RATIO	Reg NW	score = 7.46
	Reg W	score = 7.36
	Exc NW	score = 8.69
	Exc W	score = 8.18
	NW/W (Reg)	t = 0.69 p = 0.48737
	NW/W (Exc)	t = 2.99 p = 0.00365
	Reg/Exc (NW)	t = 6.21 p < 0.00001
	Reg/Exc (W)	t = 7.16 p < 0.00001
	NW/W	F = 80.56 p < 0.00001
	Reg/Exc	F = 7.24 p = 0.00781
	intn	F = 3.10 p = 0.08001

Table 17. *Simulated naming latencies for the Glushko non-words and their controls. For each mathematical relation is shown the mean scores and the statistical significances of the differences.*

		McCann & Besner (3 × 80 words)
LOG ERROR	Pseudohomophones	score = 5.96
	Control Non-words	score = 6.46
	Control Words	score = 5.10
	Pseudohs/NonWords	t = -1.35 p = 0.17968
	Pseudohs/Words	t = 2.72 p = 0.00732
	NonWords/Words	t = 4.76 p < 0.00001
– 1 / LOG ERROR	Pseudohomophones	score = 16.75
	Control Non-words	score = 17.75
	Control Words	score = 3.95
	Pseudohs/NonWords	t = -0.10 p = 0.91992
	Pseudohs/Words	t = 2.30 p = 0.02257
	NonWords/Words	t = 1.67 p = 0.09639
LOG SUM LUCE RATIO	Pseudohomophones	score = 7.92
	Control Non-words	score = 8.00
	Control Words	score = 7.46
	Pseudohs/NonWords	t = -0.53 p = 0.59930
	Pseudohs/Words	t = 3.32 p = 0.00113
	NonWords/Words	t = 4.04 p = 0.00008

Table 18. *Simulated naming latencies for the McCann & Besner pseudohomophones, control non-words and control words. For each mathematical relation is shown the mean scores and the statistical significances of the differences.*

	Regular Words	Non-Words	Exception Words	Regularization Errors
Scaling	91.7	95.3	54.2	77.3
Reduction	89.6	86.0	68.7	80.0
Clipping	91.7	79.1	81.2	11.1
Noise	91.7	88.4	47.9	72.0
Connections	91.7	83.7	39.6	69.0
Units	91.7	93.0	47.9	64.0

Table 19. *The percentage of exception word errors that are regularizations for our six types of network damage. Each case corresponds to the plotted run at the nearest data point to a regular word performance of 91.7%.*

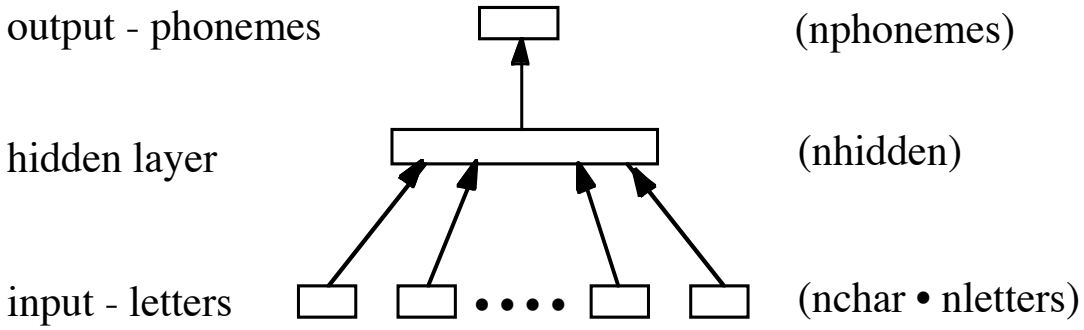

Figure 1. *The Basic Model Network Architecture with one output phoneme per word presentation.*

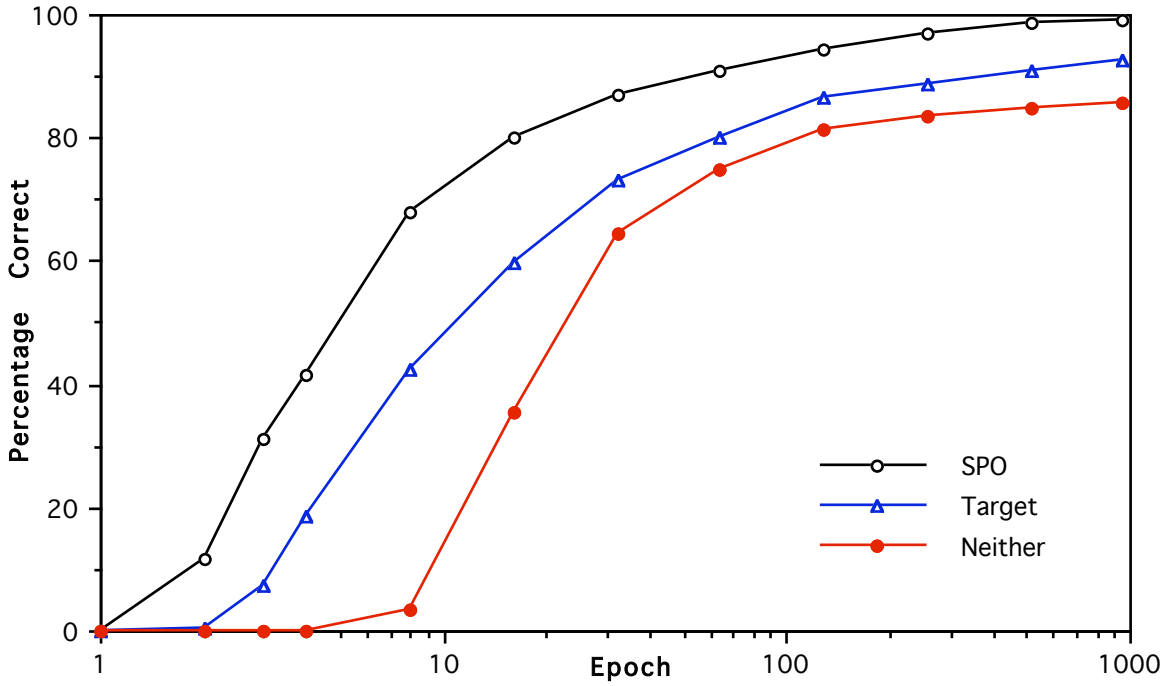

Figure 2. *Comparison of the training data learning curves for two techniques used to prevent the output activations becoming stuck hard wrong. SPO refers to using a Sigmoid Prime Offset. Target means setting the targets to 0.1 and 0.9 instead of 0.0 and 1.0. In each case we have 40 hidden units and a window size of 7 characters.*

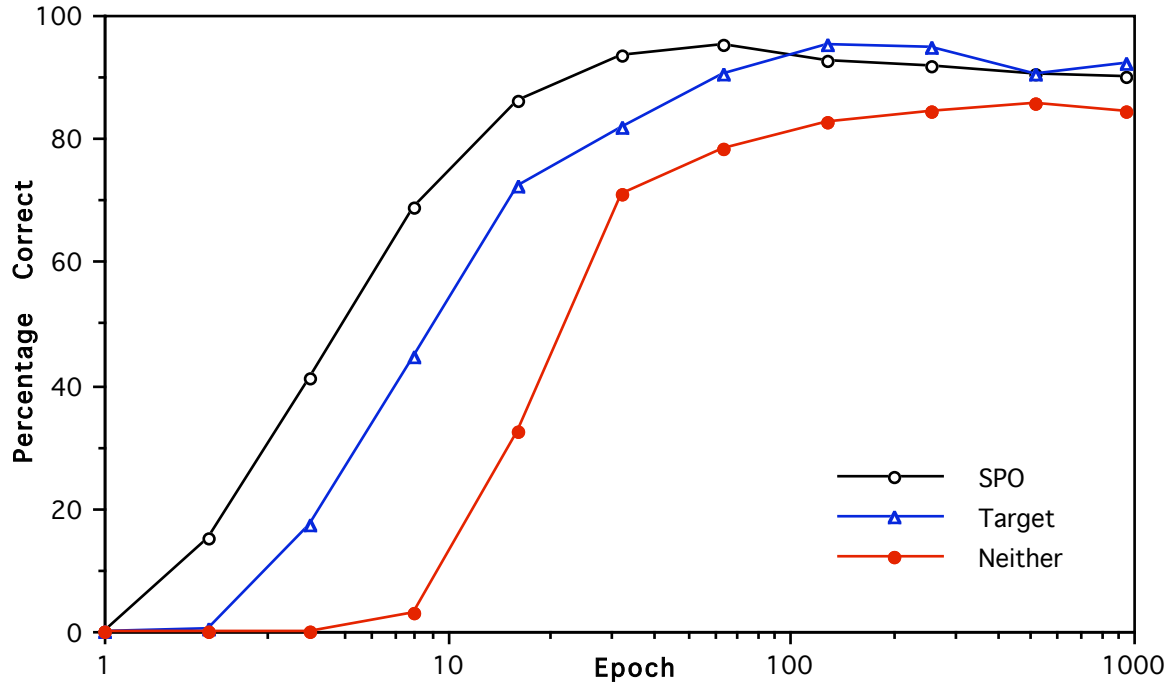

Figure 3. *Comparison of the generalization performance curves for two techniques used to prevent the output activations becoming stuck hard wrong. SPO refers to using a Sigmoid Prime Offset. Target means setting the targets to 0.1 and 0.9 instead of 0.0 and 1.0. In each case we have 40 hidden units and a window size of 7 characters.*

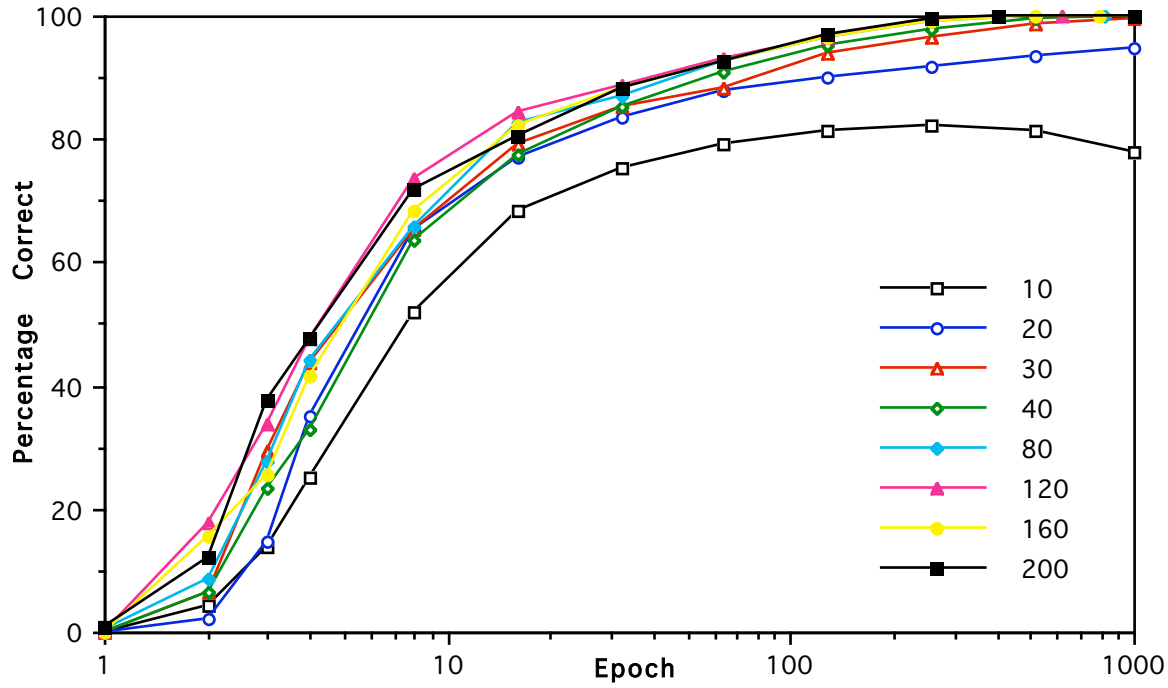

Figure 4. *The variation of the training data learning curves with the number of hidden units. Each network has a window size of 13 characters, errcrit = 0.01 and a single context flag to resolve the homograph ambiguities.*

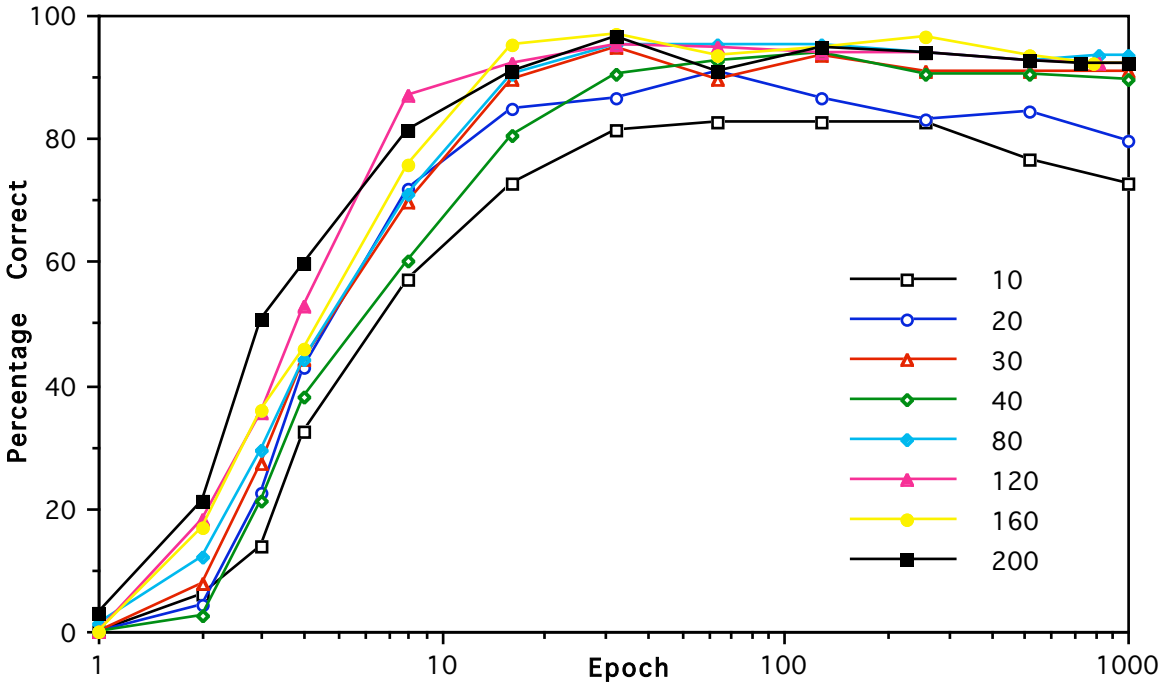

Figure 5. *The variation of the generalization performance curves with the number of hidden units. Each network has a window size of 13 characters, errcrit = 0.01 and a single context flag to resolve the homograph ambiguities.*

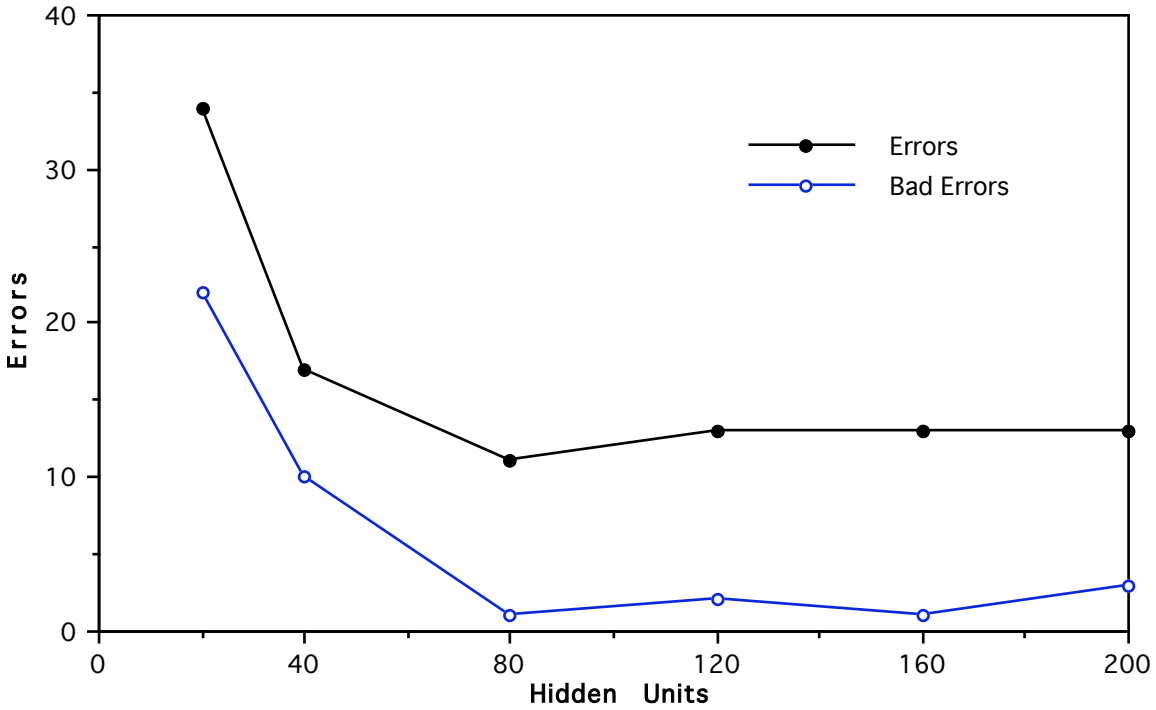

Figure 6 *The variation of the number of non-word errors with the number of hidden units. Each network has a window size of 13 characters, errcrit = 0.01 and a single context flag to resolve the homograph ambiguities.*

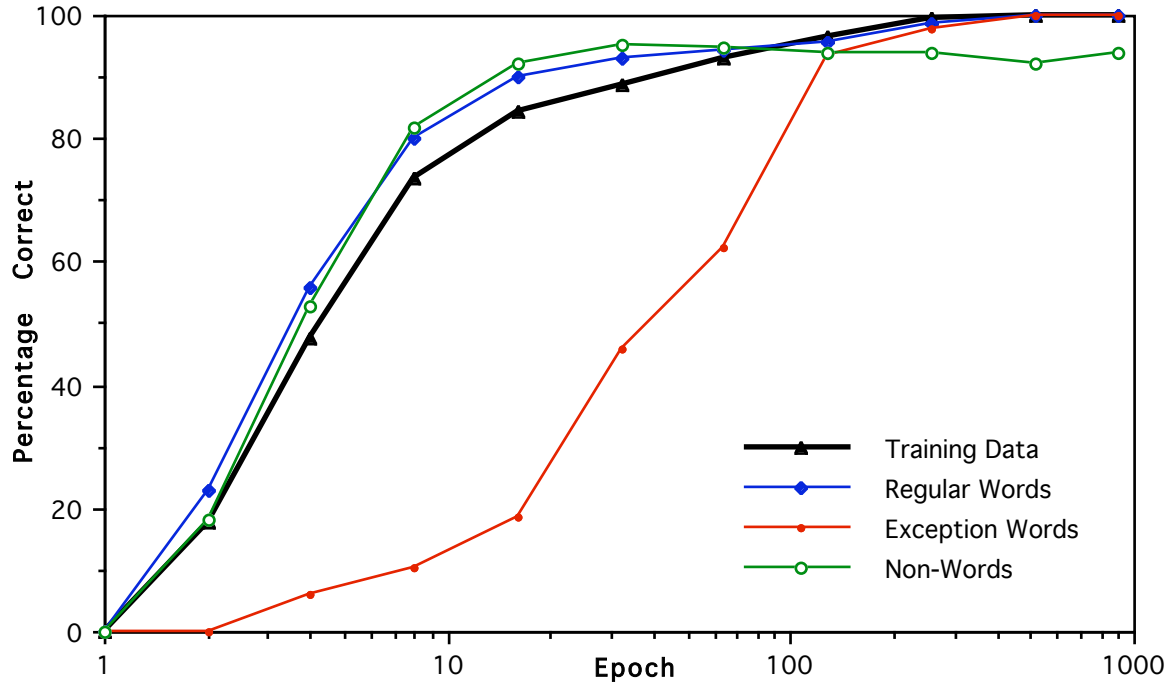

Figure 7. *The learning curves for a typical run of the basic network with 120 hidden units, a window size of 13 characters, errcrit = 0.01 and a single context flag to resolve the homograph ambiguities.*

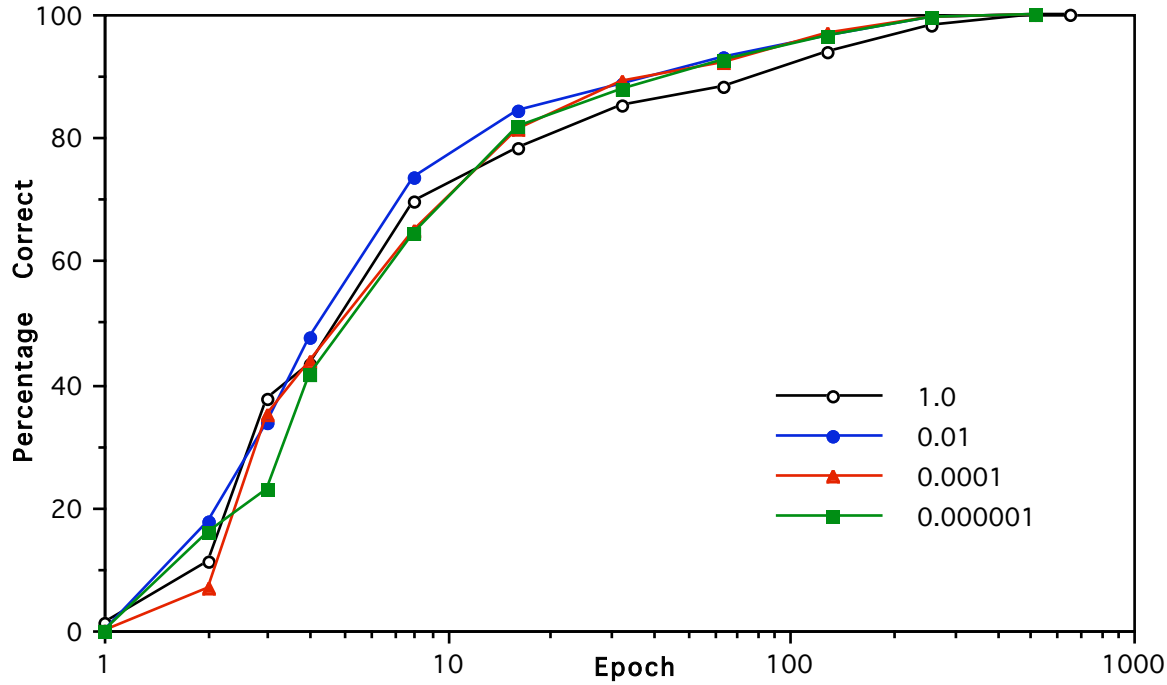

Figure 8. *The variation of the training data learning curves with the over-learning parameter errcrit. Each network has a window size of 13 characters, 120 hidden units and a single context flag to resolve the homograph ambiguities.*

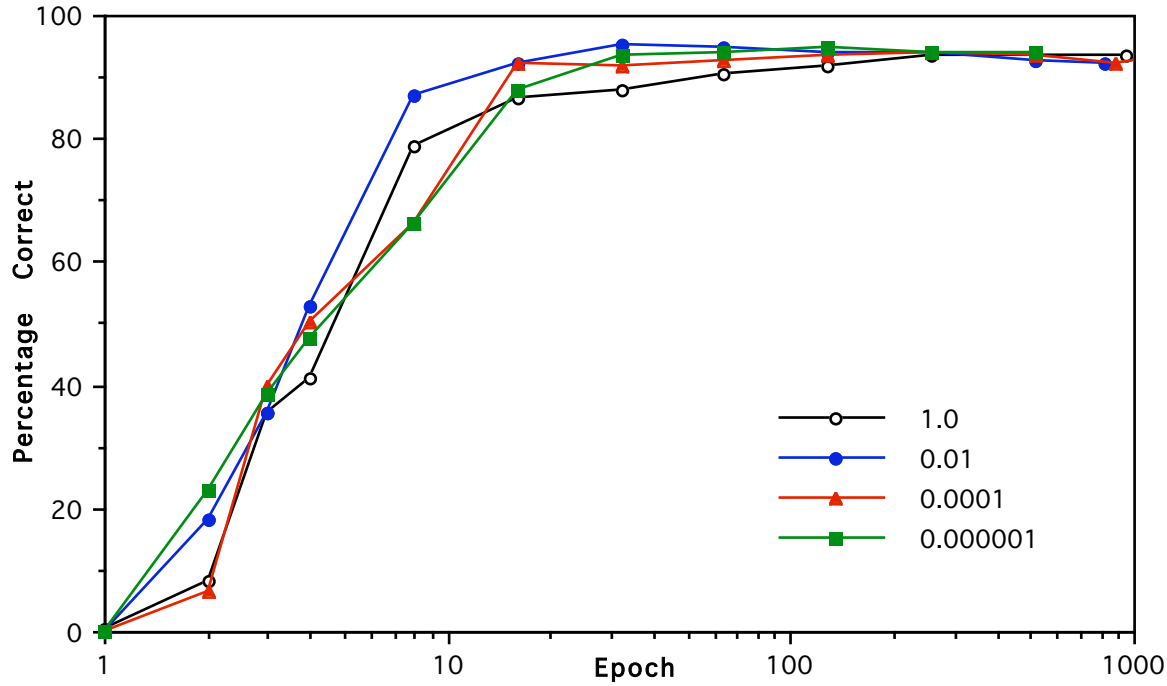

Figure 9. *The variation of the generalization performance with the over-learning parameter errcrit. Each network has a window size of 13 characters, 120 hidden units and a single context flag to resolve the homograph ambiguities.*

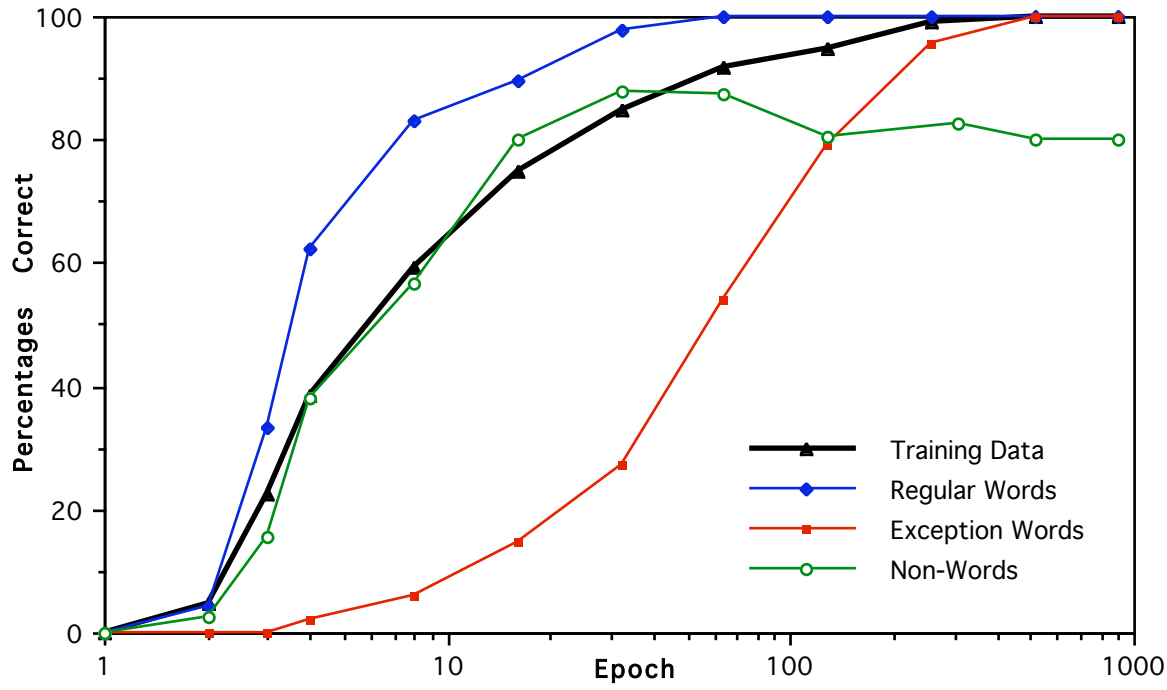

Figure 10. *The learning curves for a typical run of the basic network with no explicit blanks, 120 hidden units, a window size of 13 characters, errcrit = 0.01 and a single context flag to resolve the homograph ambiguities.*

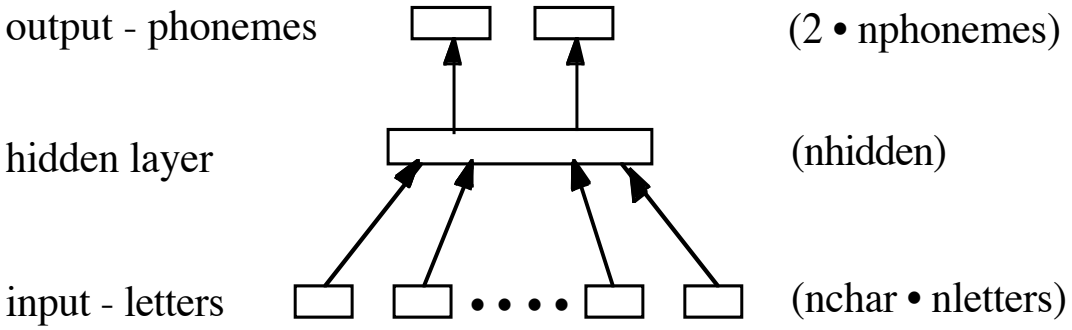

Figure 11. *The Basic Model Network Architecture with two output phonemes per word presentation.*

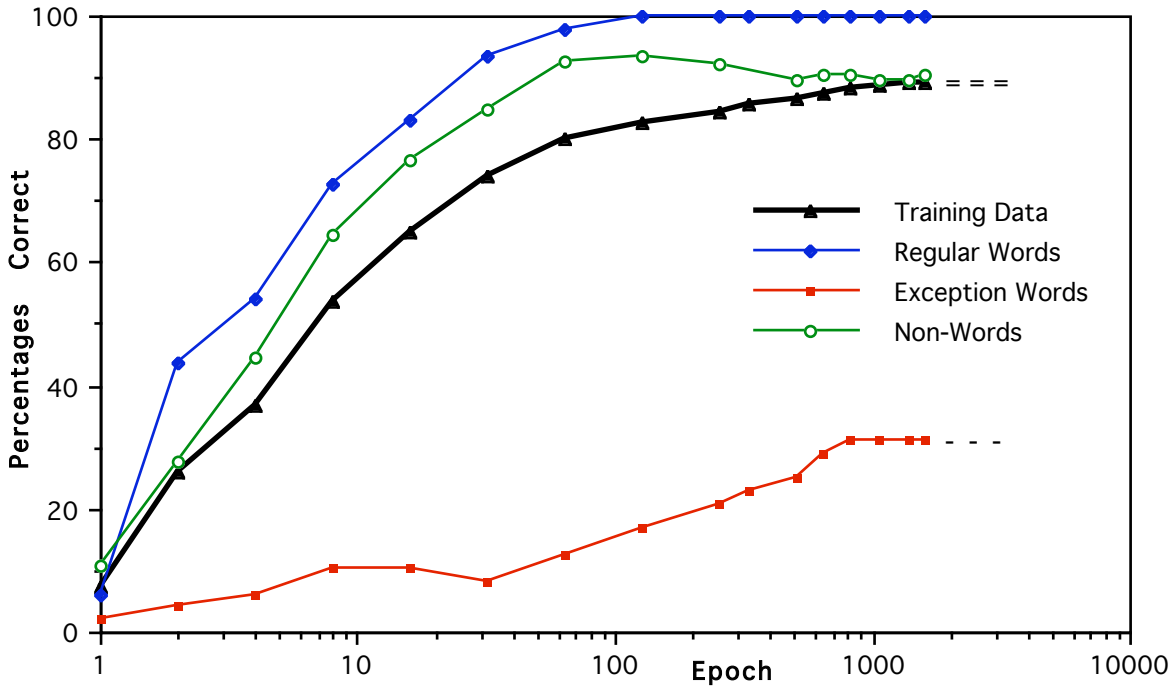

Figure 12. *The learning curves for a typical run of a network with no hidden units, a window size of 13 characters, errcrit = 0.01 and a single context flag to resolve the homograph ambiguities. The data points after epoch 512 are averages over several epochs since there is much variation from epoch to epoch.*

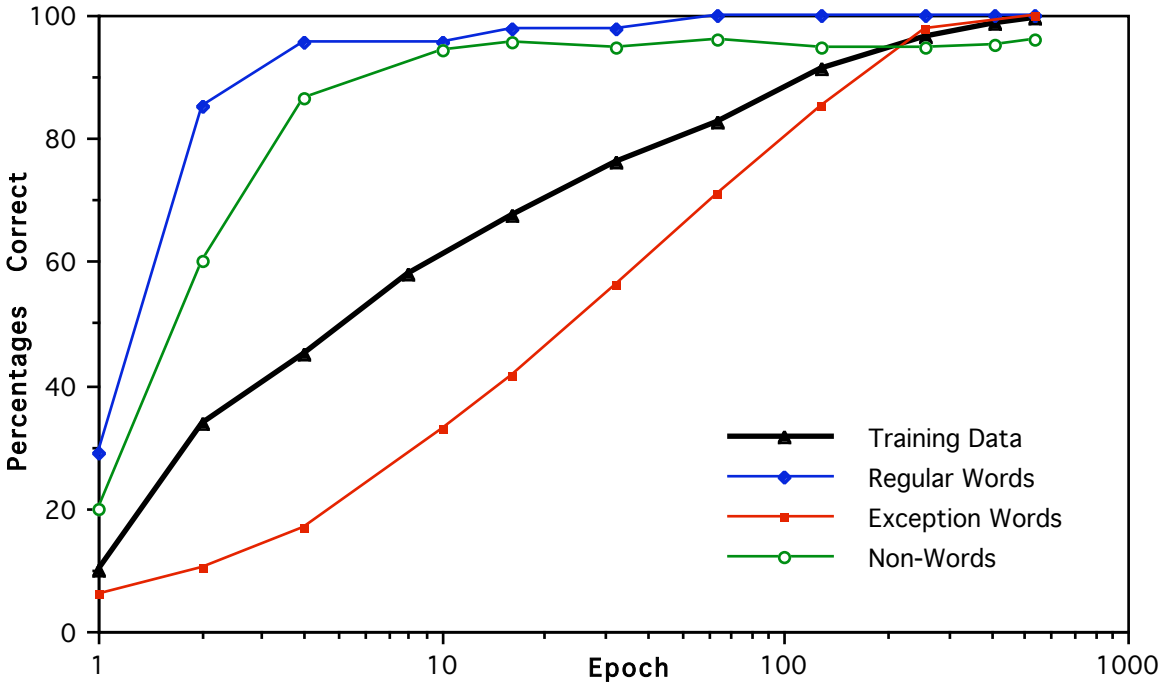

Figure 13. *The learning curves for a typical run of the basic network with the extended training data set of 13891 words including many multi-syllabic words but no homographs. The network had 160 hidden units, a window size of 17 characters and errcrit = 0.01.*

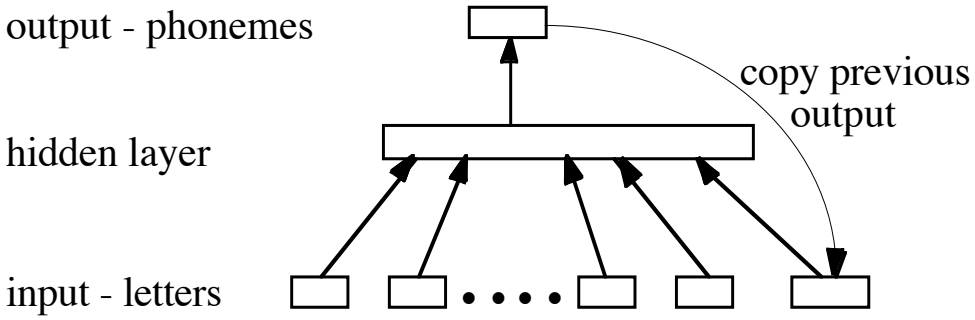

Figure 14. *The Basic Model Network Architecture with one output phoneme per word presentation and recurrent connections.*

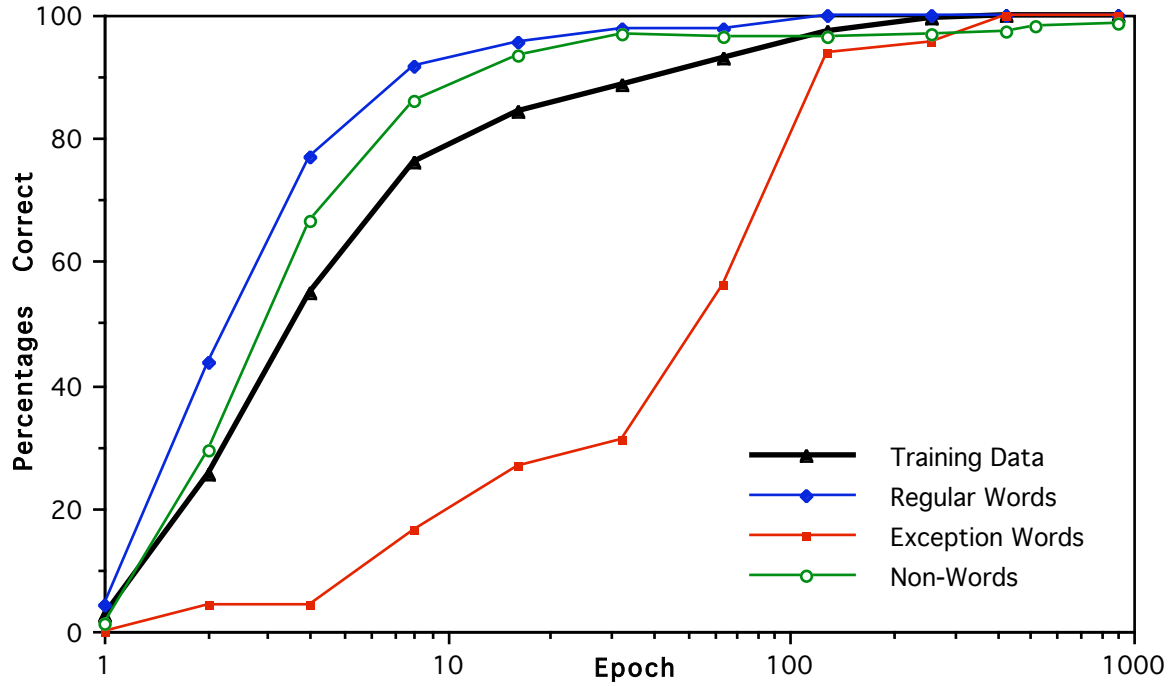

Figure 15. *The learning curves for a typical run of the basic two phoneme output model with 300 hidden units, a window size of 13 characters, errcrit = 0.0 and a single context flag to resolve the homograph ambiguities.*

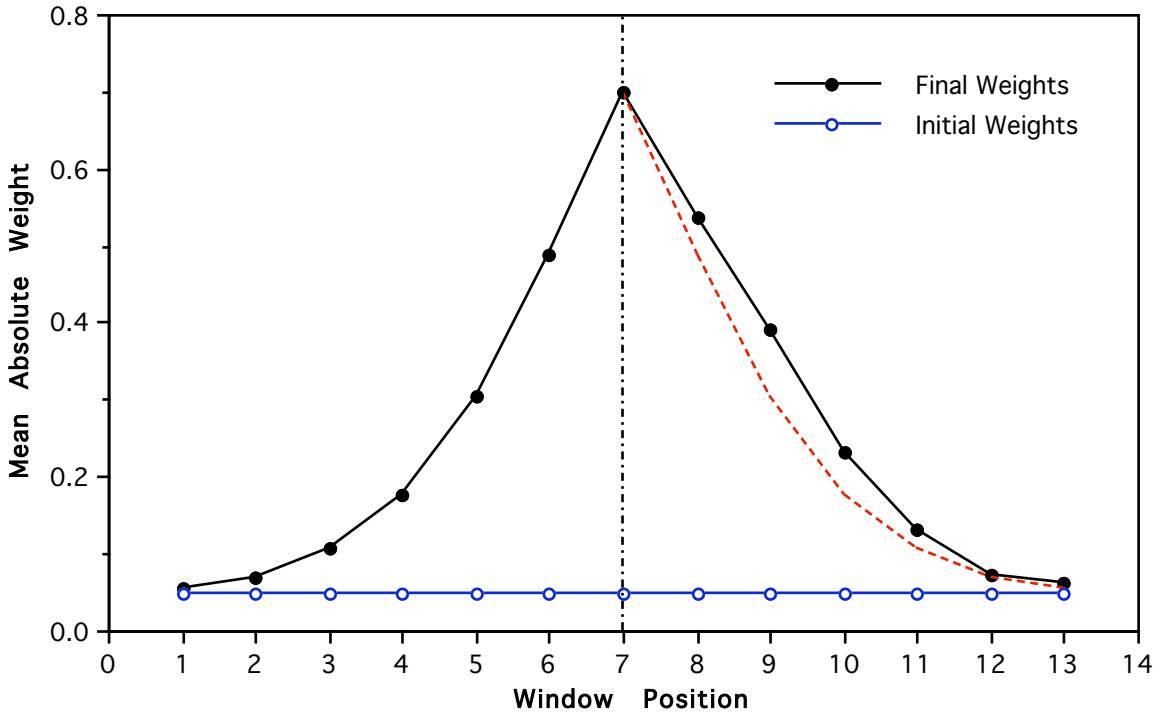

Figure 16. *The final distribution of weights for a typical run of the basic two phoneme output model with 300 hidden units, a window size of 13 characters, errcrit = 0.0 and a single context flag to resolve the homograph ambiguities.*

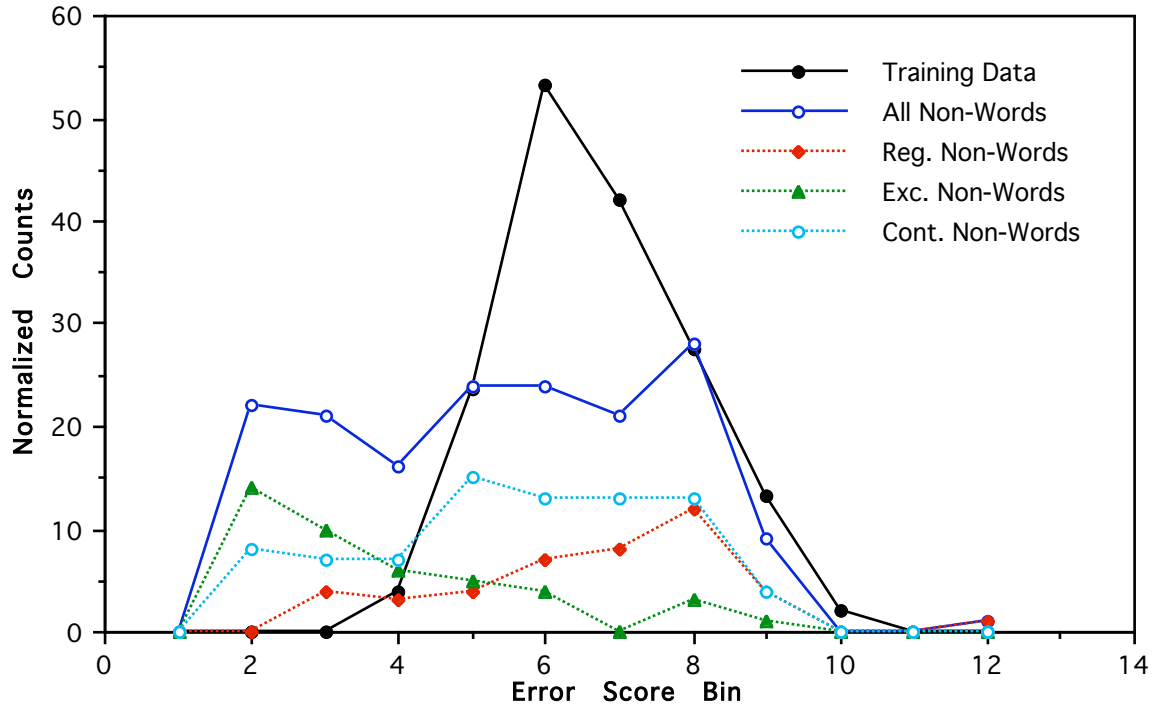

Figure 17. *The final distribution of output error scores for the training data and the various non-word sets. The bins follow a logarithmic scale with the highest numbered bins corresponding to the lowest error scores.*

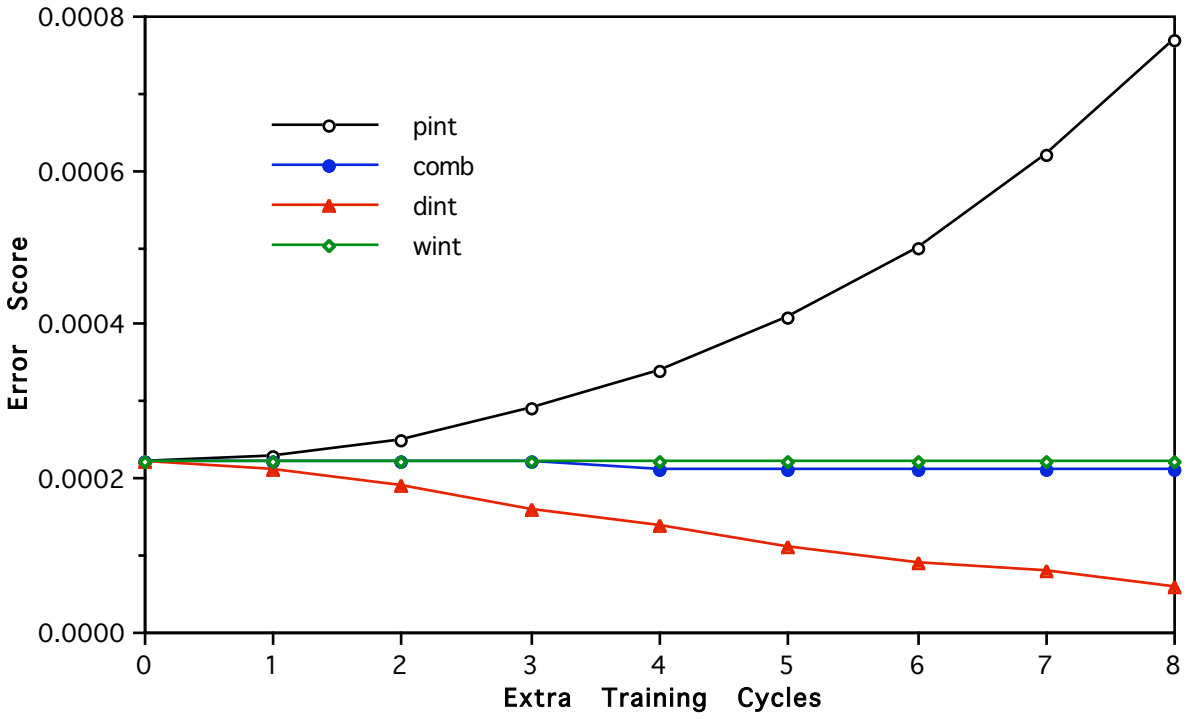

Figure 18. *The effect on the output error score for the word 'tint' of extra network training on the words 'pint', 'comb', 'dint' and 'wint'.*

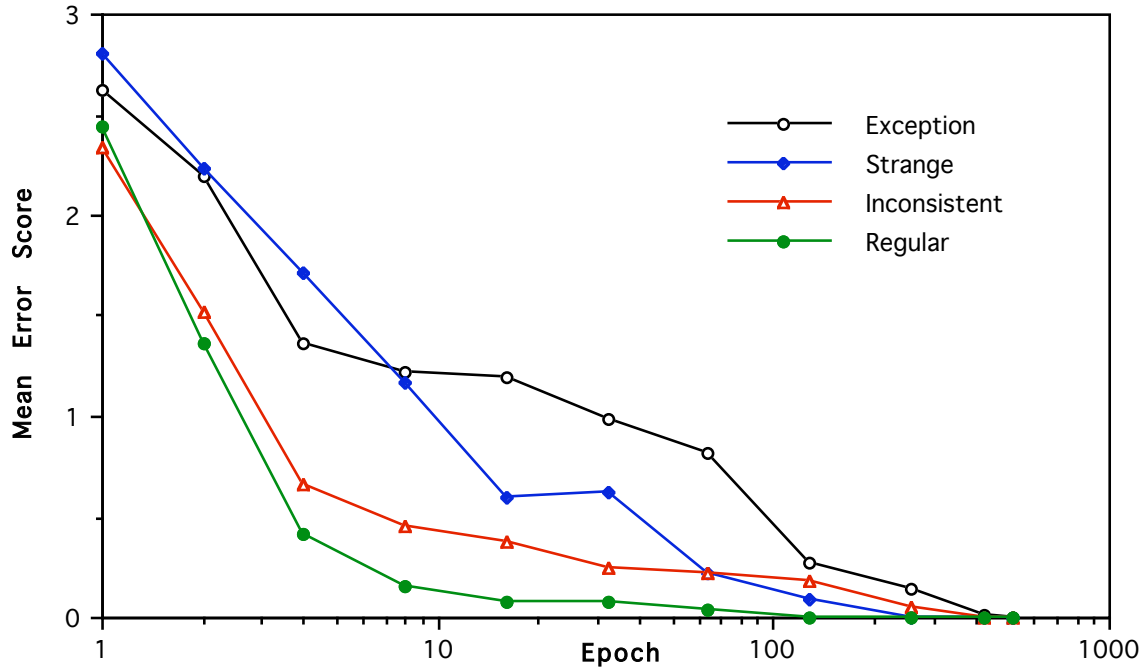

Figure 19. The evolution during training of the output activation error scores for the low frequency exception, strange, regular inconsistent and regular words.

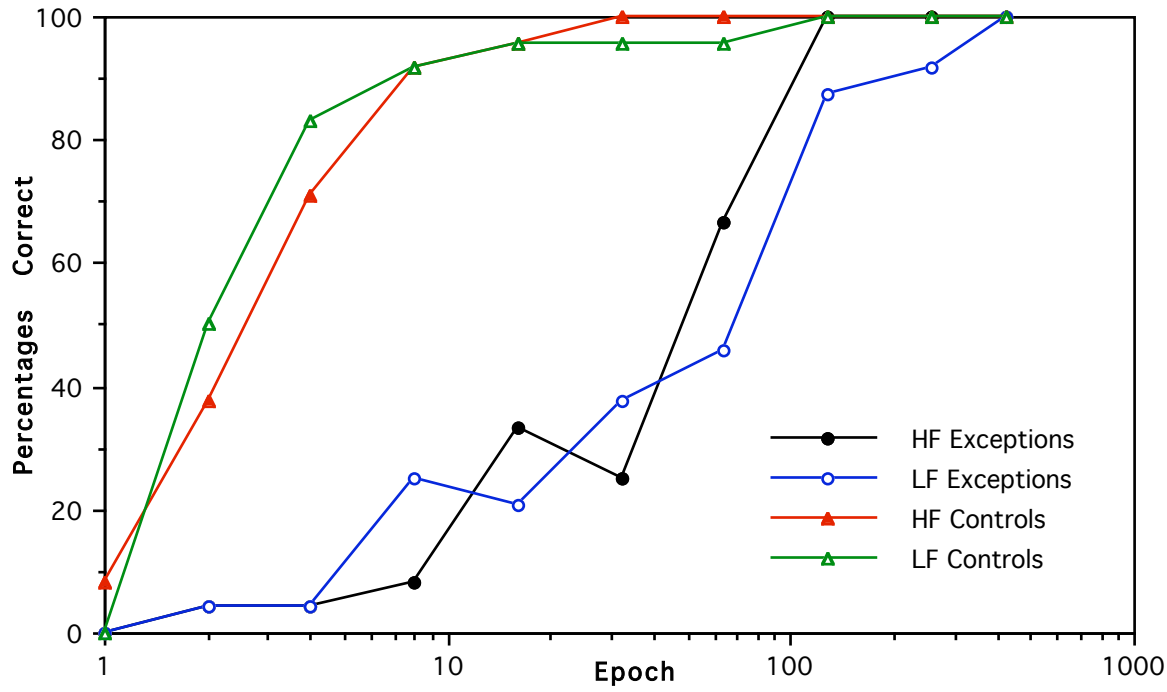

Figure 20. *The learning curves of the (Taraban & McClelland) high and low frequency exception words and the corresponding regular control words.*

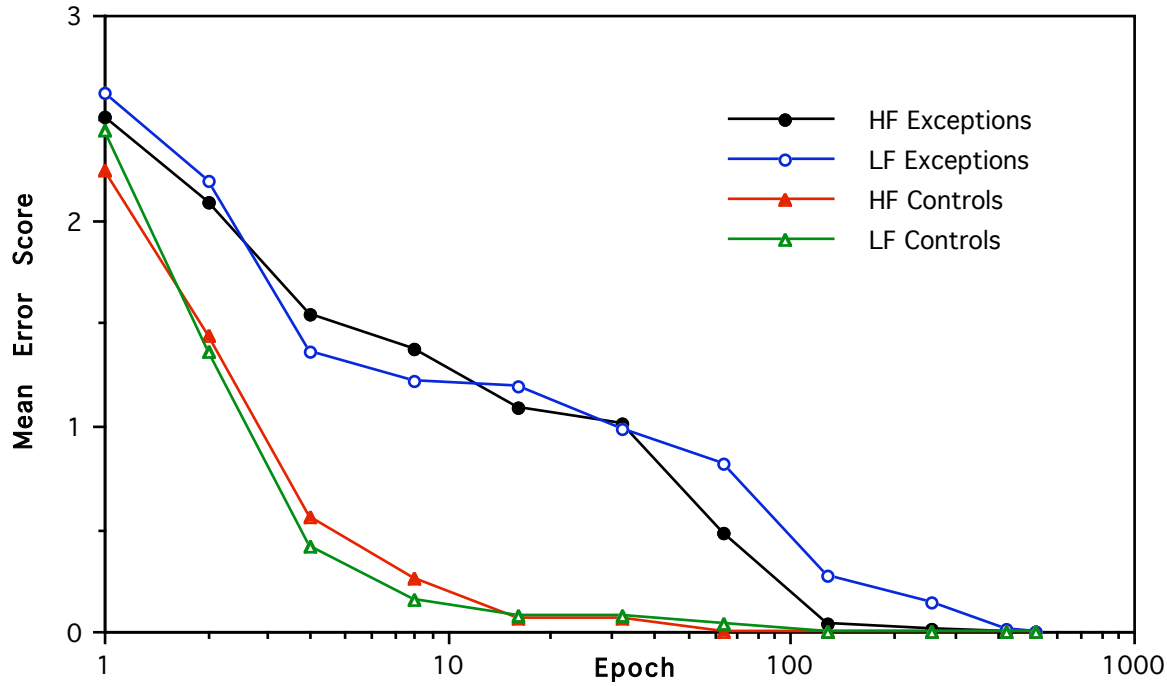

Figure 21. *The evolution during training of the output activation error scores for the (Taraban & McClelland) high and low frequency exception words and the corresponding regular control words.*

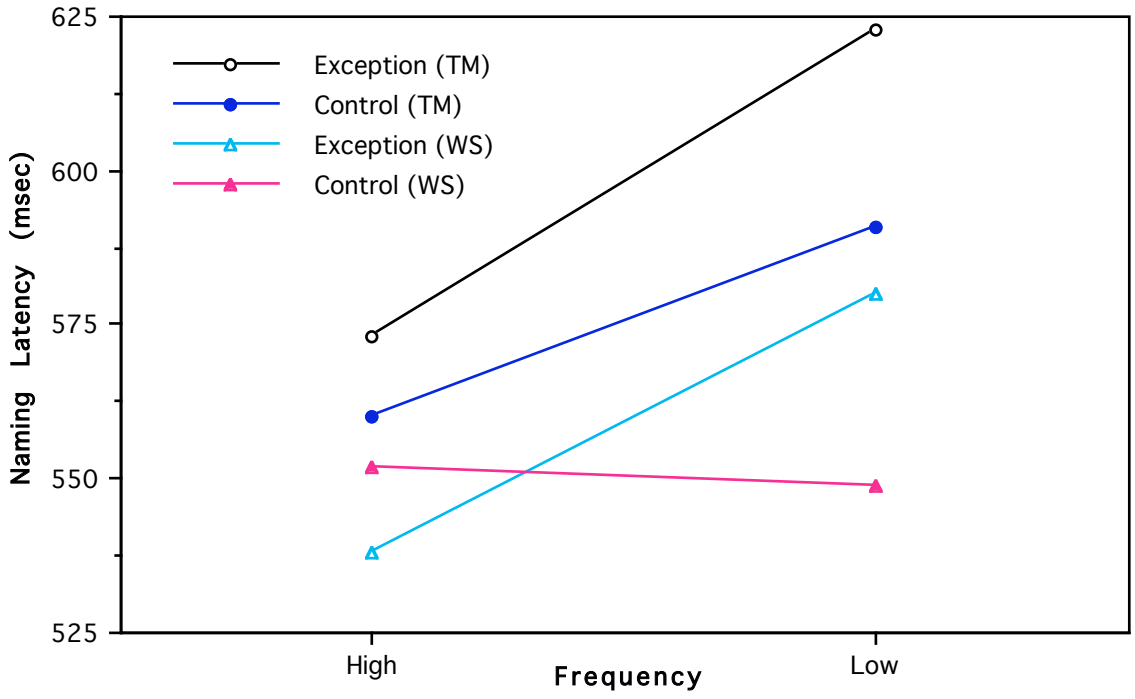

Figure 22. *The naming latencies for humans on the Taraban & McClelland and Waters & Seidenberg high and low frequency exception words and the corresponding regular control words.*

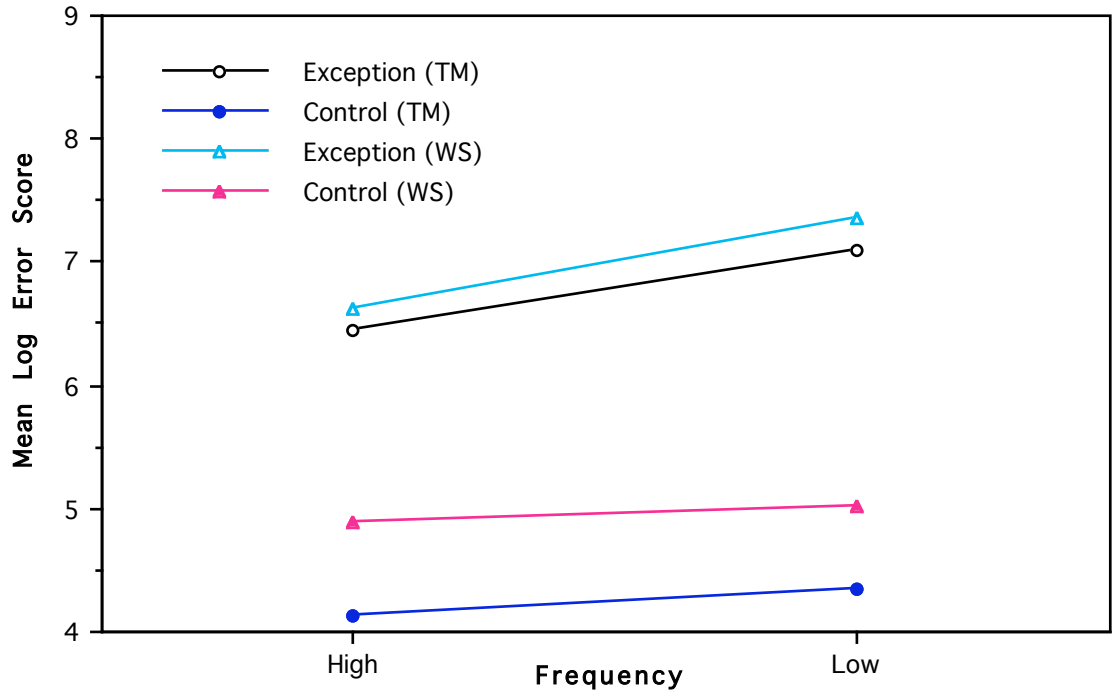

Figure 23. *The network's mean log error scores on the Taraban & McClelland and Waters & Seidenberg high and low frequency exception words and the corresponding regular control words.*

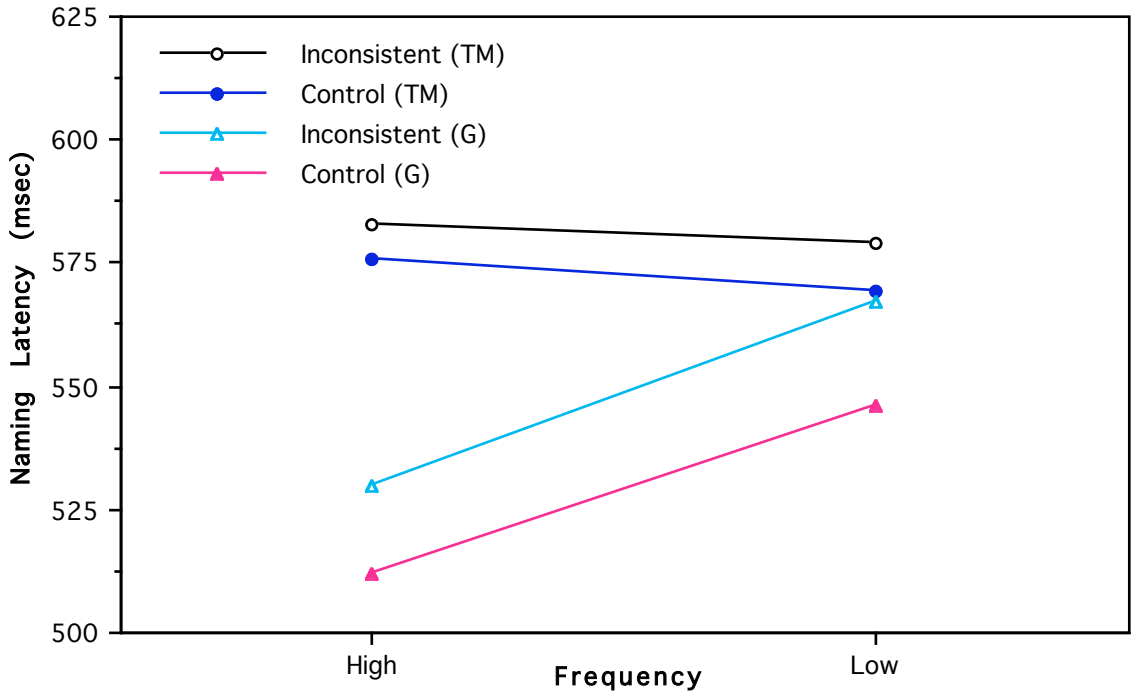

Figure 24. *The naming latencies for humans on the Taraban & McClelland and Glushko high and low frequency regular inconsistent words and the corresponding regular control words.*

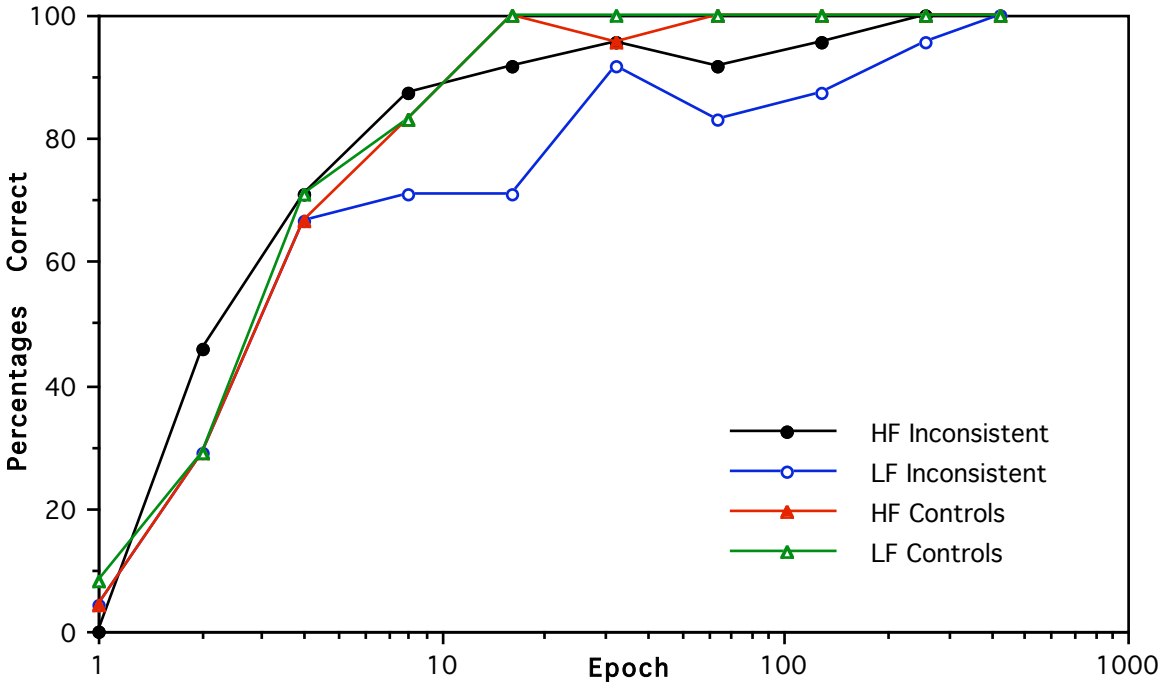

Figure 25. *The learning curves of the (Taraban & McClelland) high and low frequency regular inconsistent words and the corresponding regular control words.*

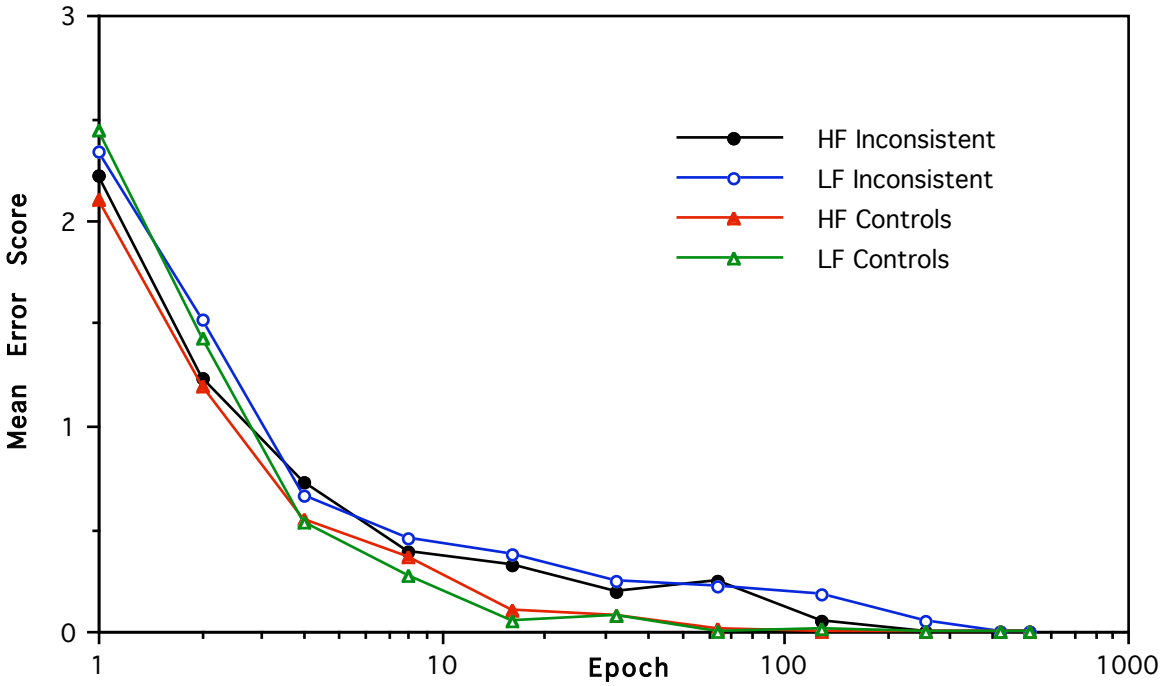

Figure 26. *The evolution during training of the output activation error scores for the (Taraban & McClelland) high and low frequency regular inconsistent words and the corresponding regular control words.*

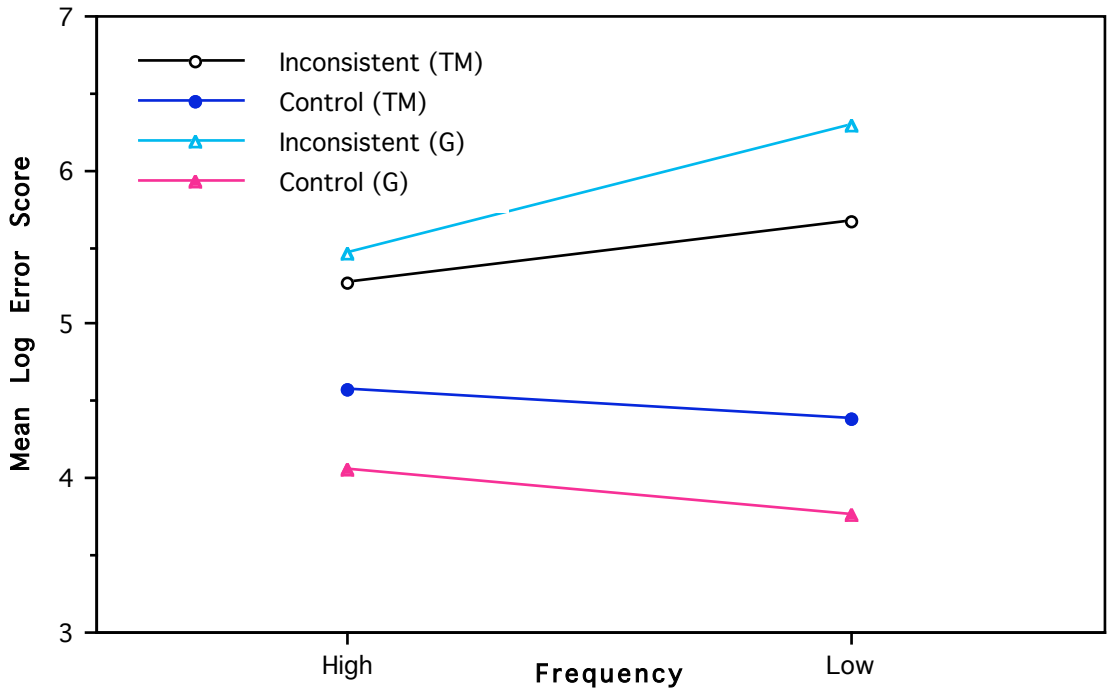

Figure 27. *The network's final mean log error scores on the Taraban & McClelland and Glushko high and low frequency regular inconsistent words and the corresponding regular control words.*

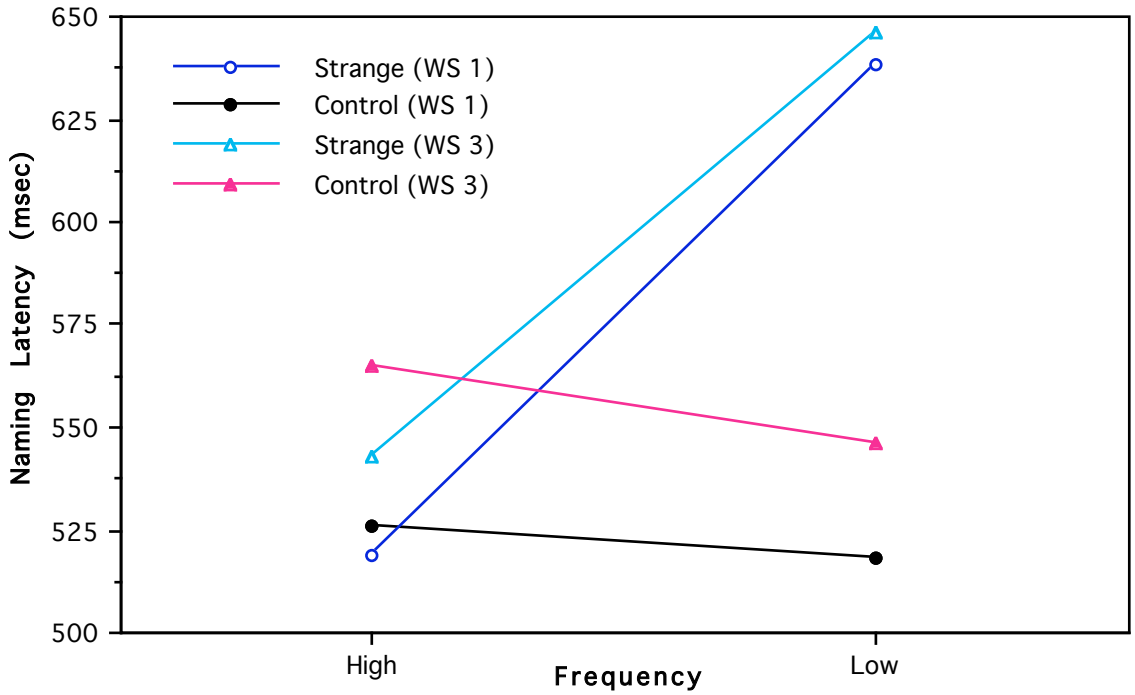

Figure 28 *The naming latencies for humans on the Waters & Seidenberg high and low frequency strange words and the corresponding regular control words.*

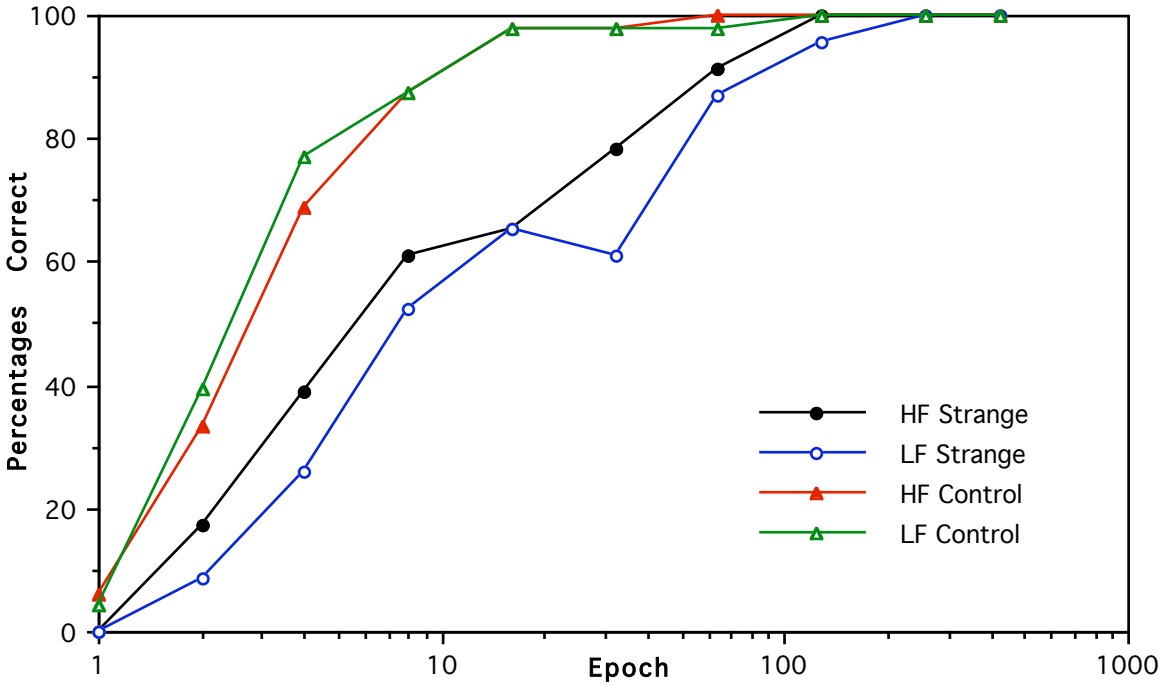

Figure 29. *The learning curves of the (Siedenberg & McClelland) high and low frequency strange words and the corresponding regular control words.*

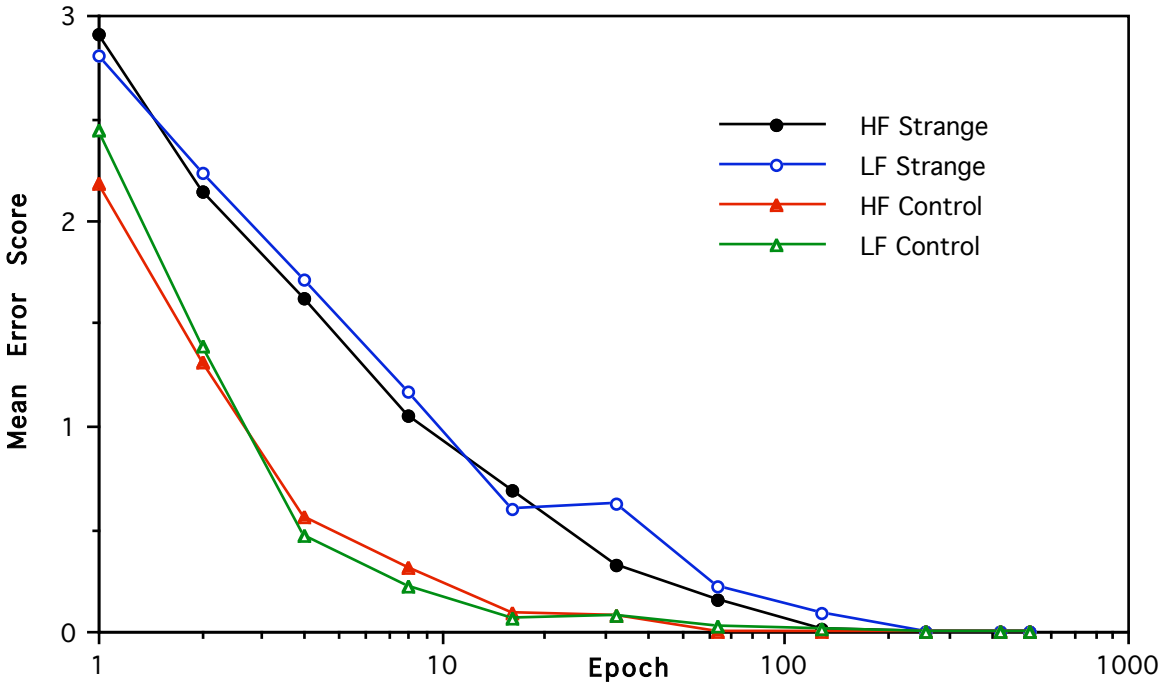

Figure 30. *The evolution during training of the output activation error scores for the (Seidenberg & McClelland) high and low frequency strange words and the corresponding regular control words.*

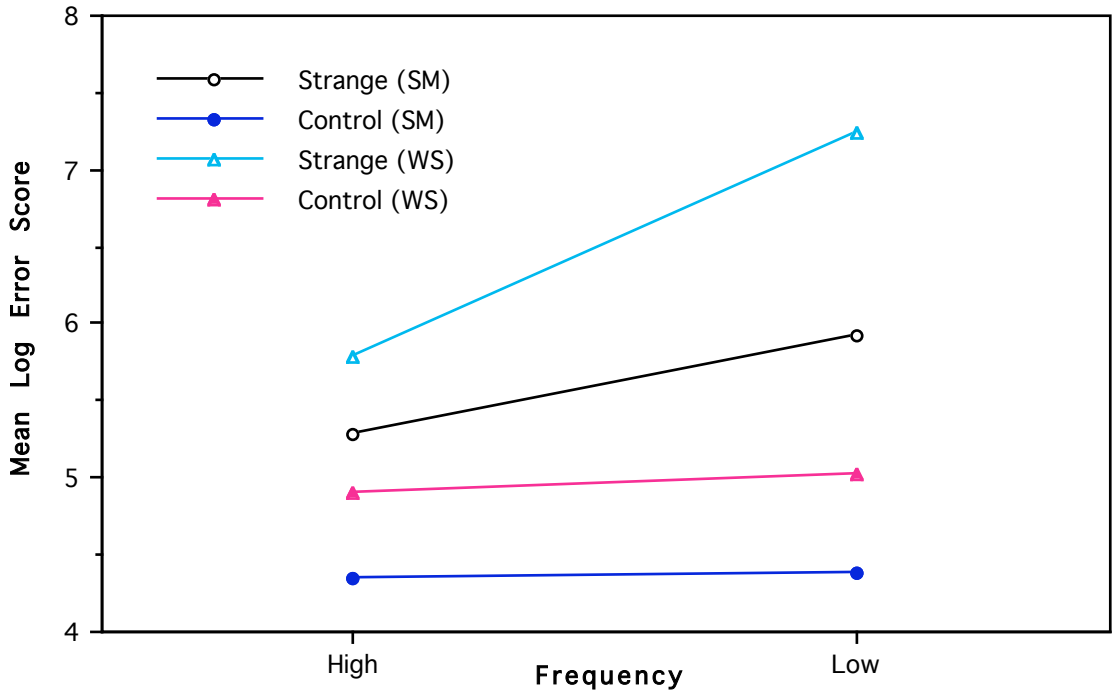

Figure 31. *The network's final mean log error scores on the Seidenberg & McClelland and Waters & Seidenberg high and low frequency strange words and the corresponding regular control words.*

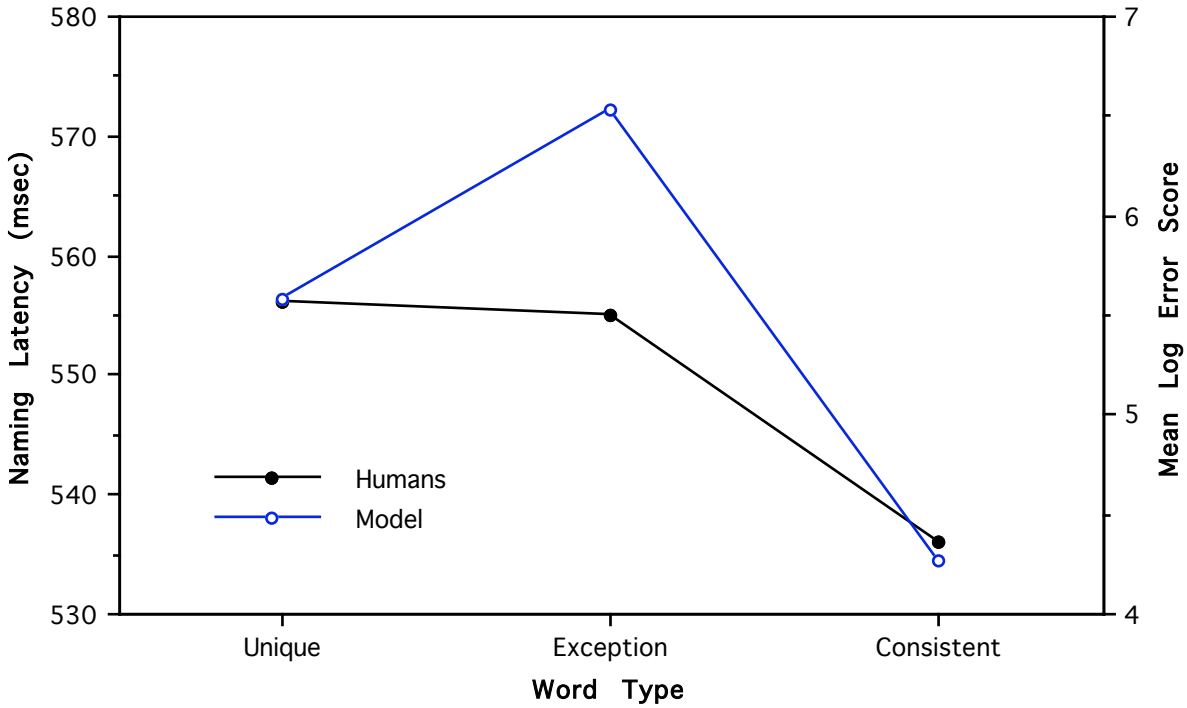

Figure 32. *The network's final mean log error scores on Brown's unique, exception and consistent words compared with the corresponding naming latencies for humans.*

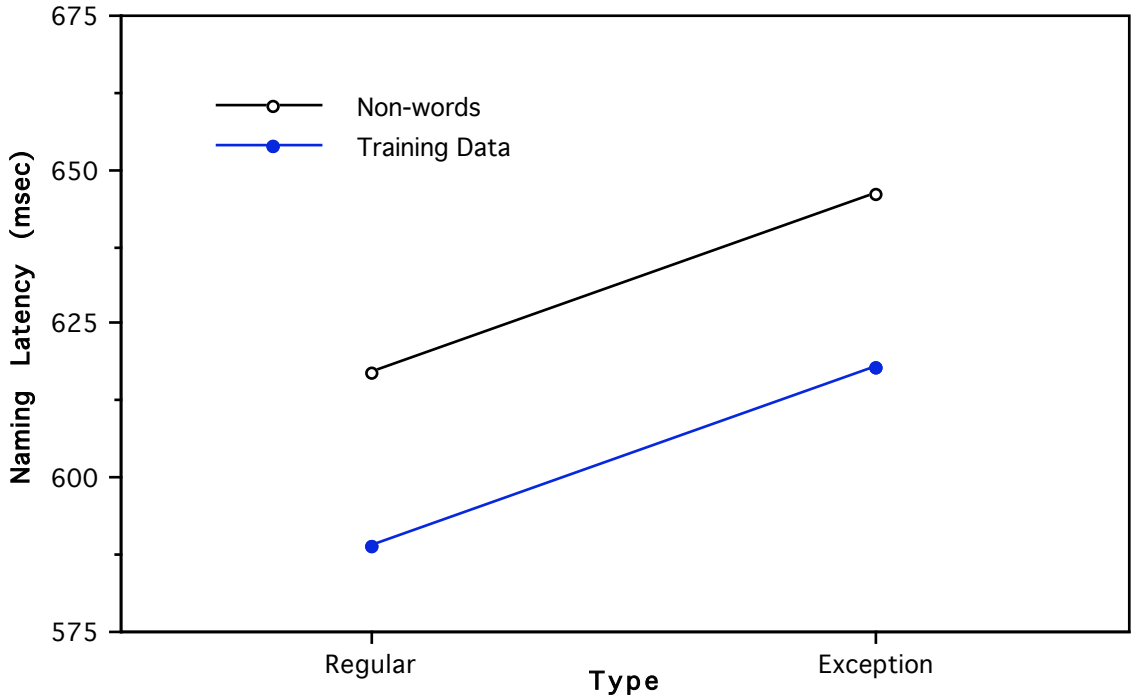

Figure 33. *The naming latencies for humans on the regular and exceptional Glushko non-words and the corresponding control words.*

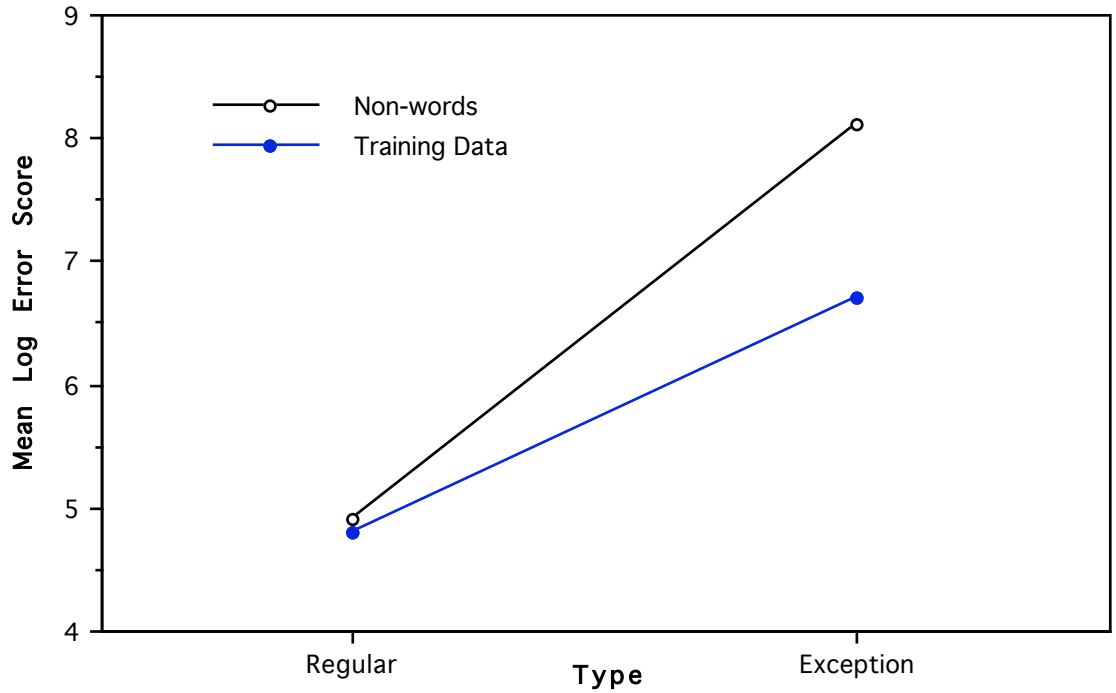

Figure 34. *The network's final mean log error scores on the regular and exceptional Glushko non-words and the corresponding control words in the training data.*

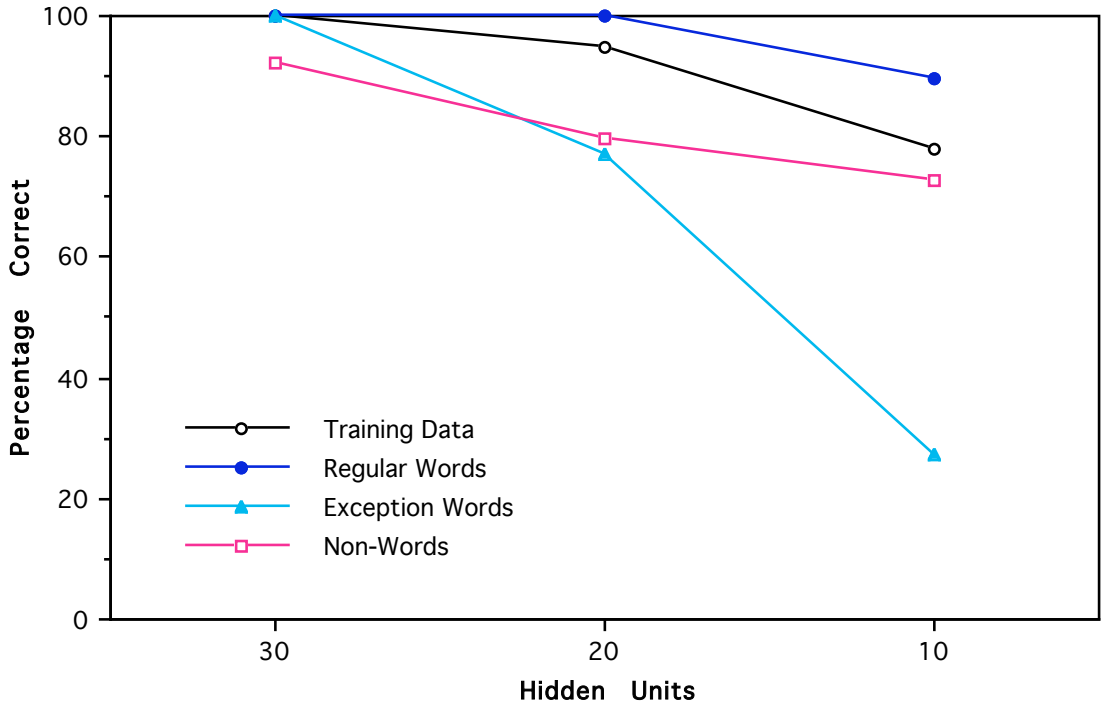

Figure 35 *The final network performance when the number of hidden units is very low. The corresponding learning curves are shown in Figures 4 and 5.*

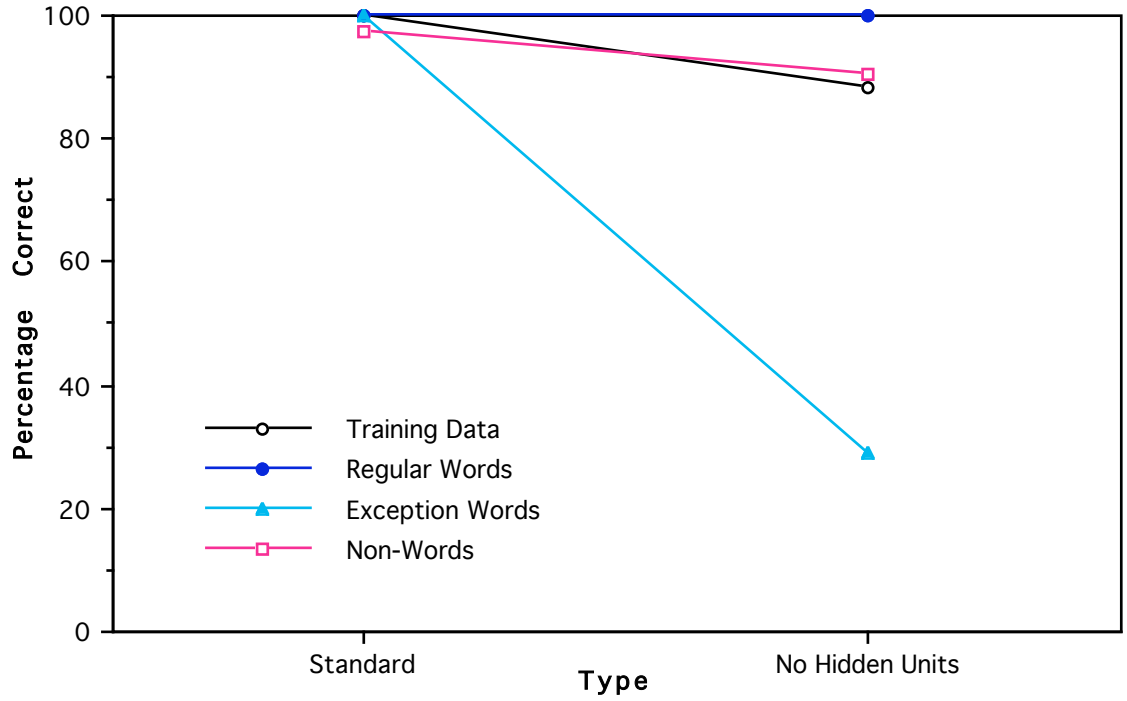

Figure 36. *The final network performance for the standard one hidden layer network and the corresponding network with only direct input to output connections. The corresponding learning curves are shown in Figures 12 and 15.*

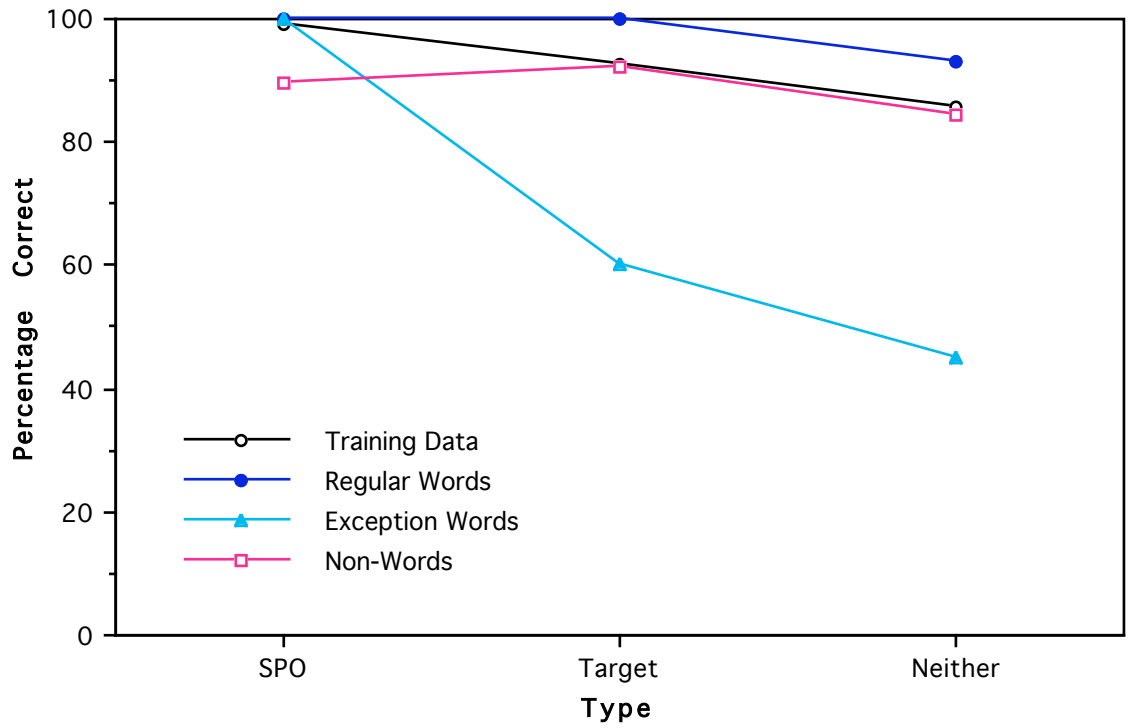

Figure 37. *The final network performance when trained with a Sigmoid Prime Offset (SPO), reduced targets of 0.1 and 0.9, and neither. The corresponding learning curves are shown in Figures 2 and 3.*

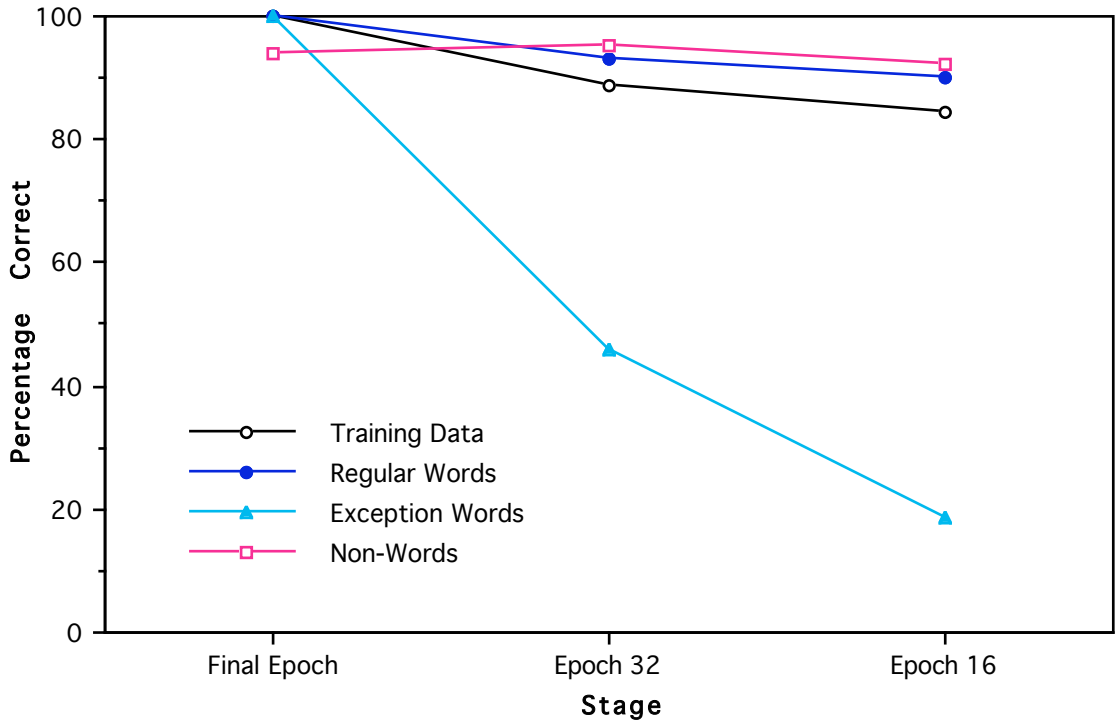

Figure 38. *The network's performance at three stages during training. The full learning curves are shown in Figure 7.*

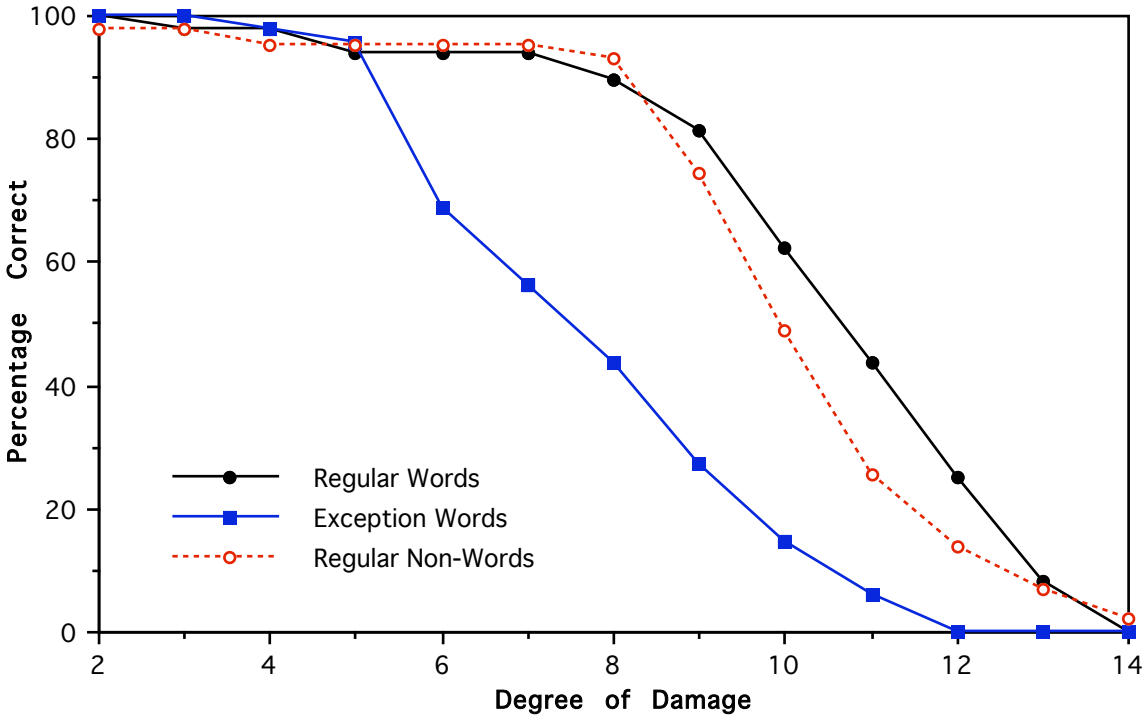

Figure 39. *The effect of damage on our errcrit = 0.0 network by global weight scaling, i.e. by the scaling of all weights by successive factors of 0.9.*

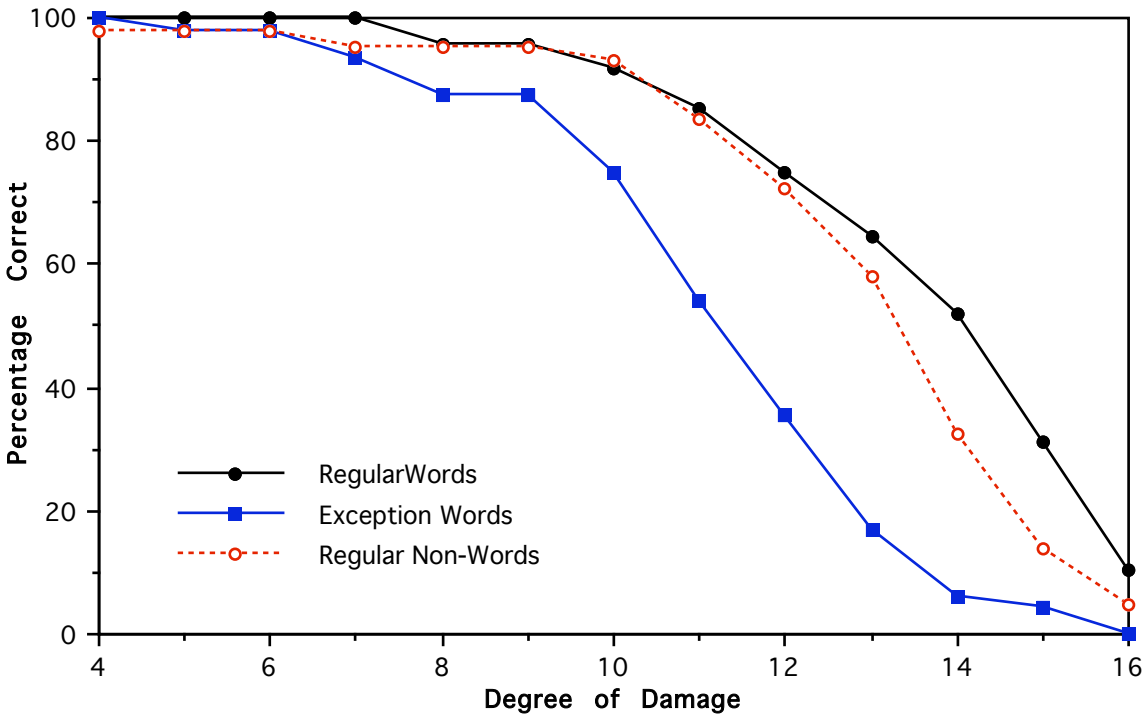

Figure 40. *The effect of damage on our errcrit = 0.0 network by global weight reduction, i.e. by the reduction of all weights by successive amounts of 0.04.*

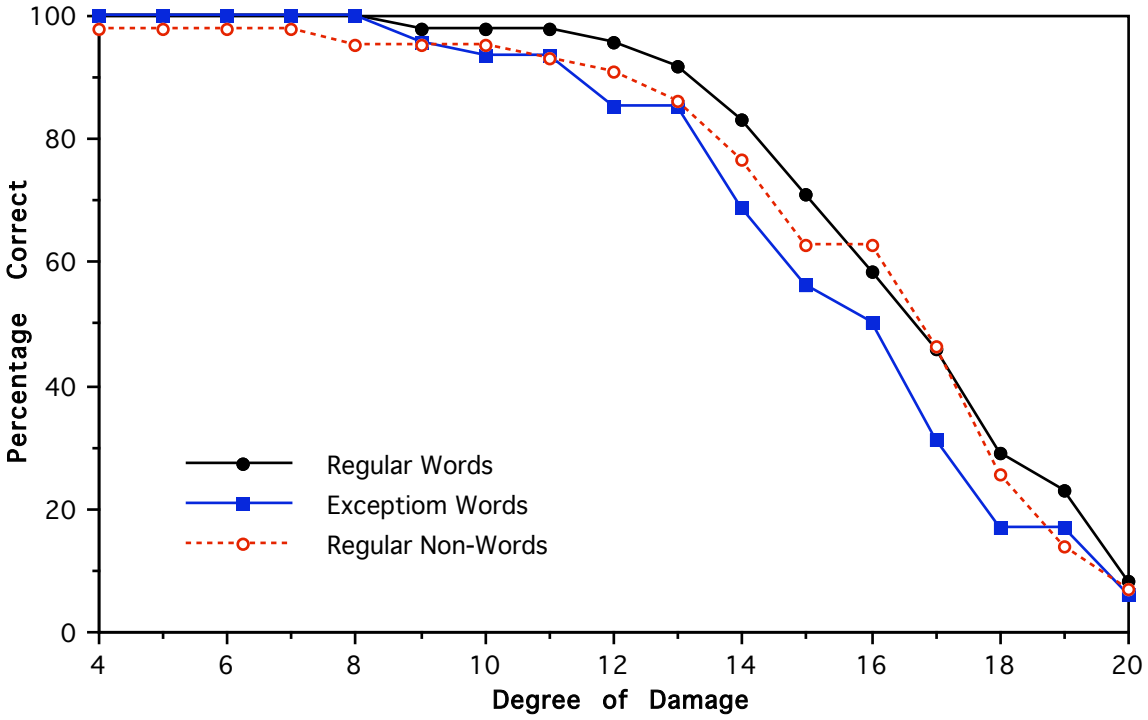

Figure 41. *The effect of damage on our errcrit = 0.0 network by global weight clipping, i.e. by the imposition of successive maximum allowable weights equal to 0.9 of the previous maximum.*

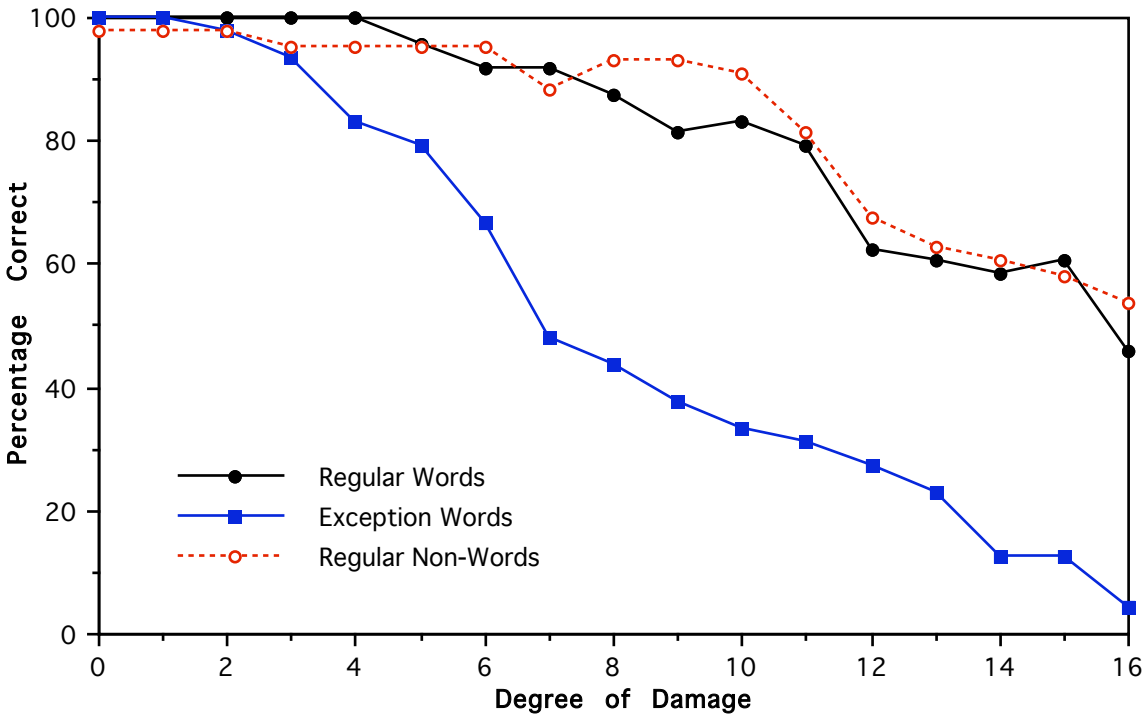

Figure 42. *The effect of damage on our errcrit = 0.0 network by adding Gaussian noise to the weights, i.e. by successive applications of random changes to the weights with zero mean and standard deviation of 0.2.*

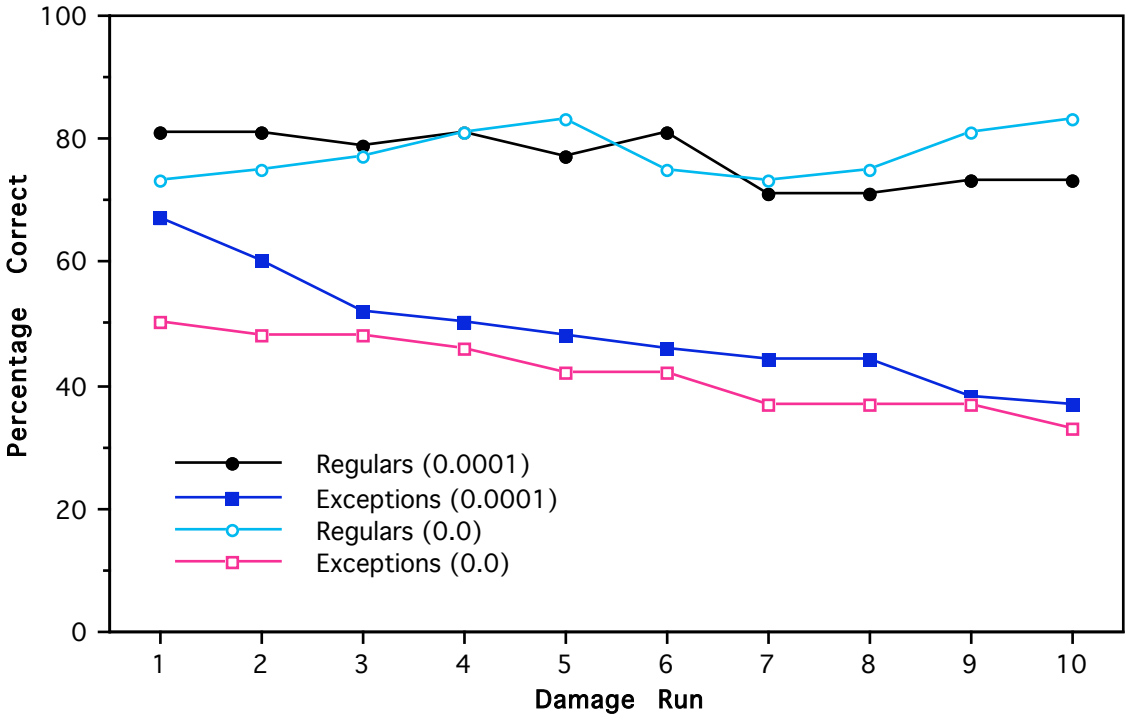

Figure 43. *An illustration of the range of variations in the size of dissociation for different sets of random noise. Plotted are the best dissociations found for each damage run of the errcrit = 0.0 and 0.0001 networks with the regular word performance better than 70%. For clarity the runs are ordered according to the exception word performances.*

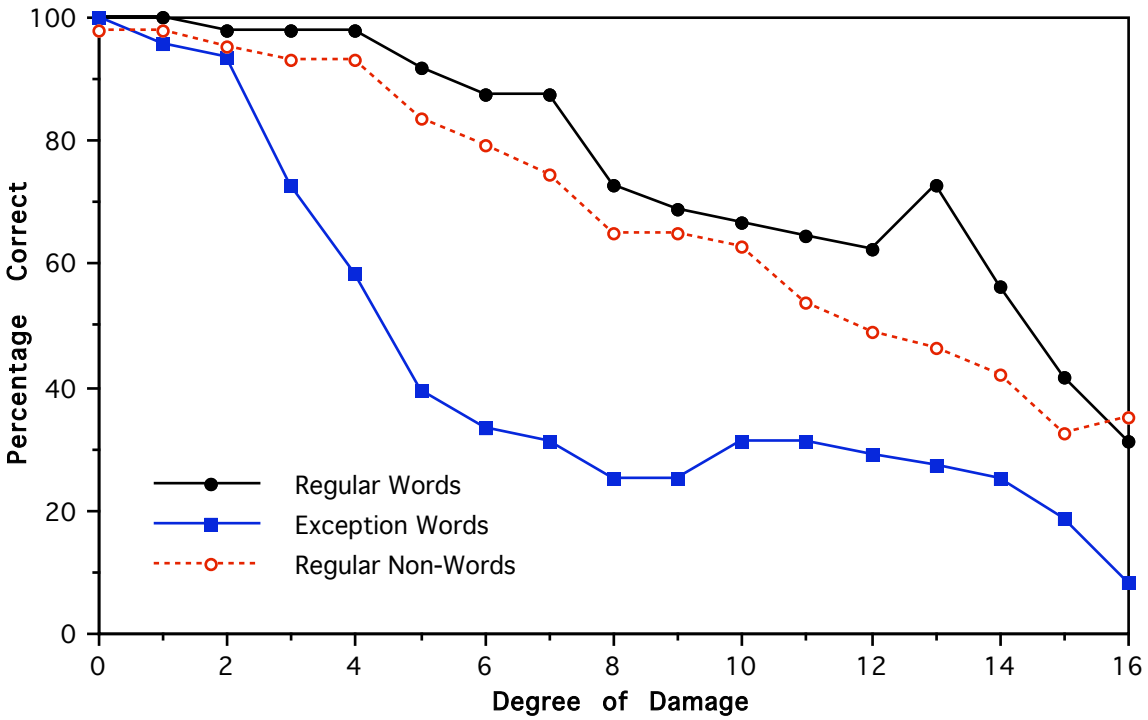

Figure 44 *The effect of damage on our errcrit = 0.0 network by the removal of connections, i.e. by the successive removal of random sets of 6600 connections (out of the total set of ~132000 connections).*

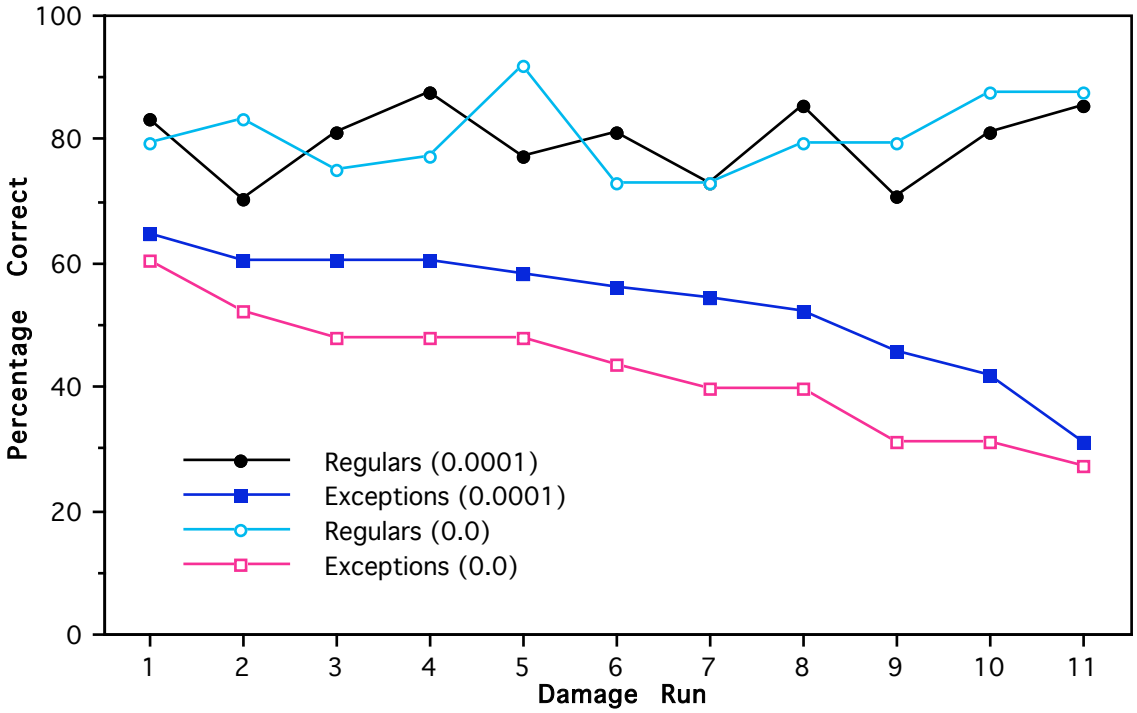

Figure 45. *An illustration of the range of variations in the size of dissociation for different sets of removed connections. Plotted are the best dissociations found for each damage run of the errcrit = 0.0 and 0.0001 networks with the regular word performance better than 70%.*

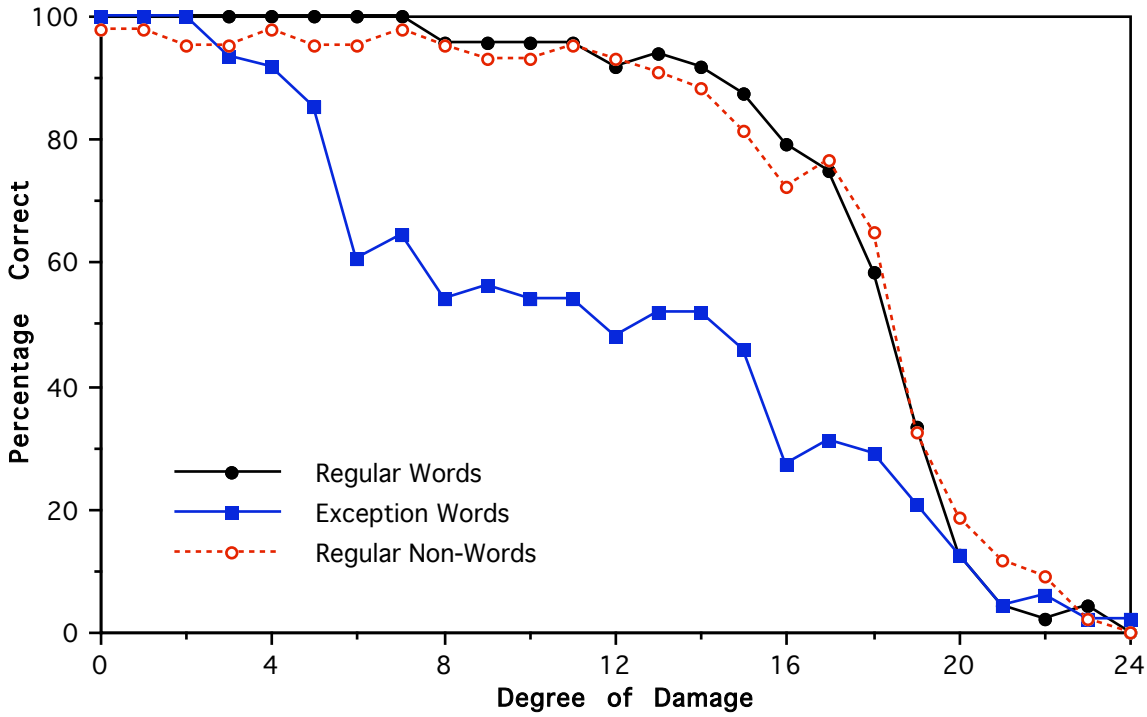

Figure 46 *The effect of damage on our errcrit = 0.0 network by the removal of hidden units, i.e. by the successive removal of random sets of ten hidden units (out of the total set of 300 hidden units).*

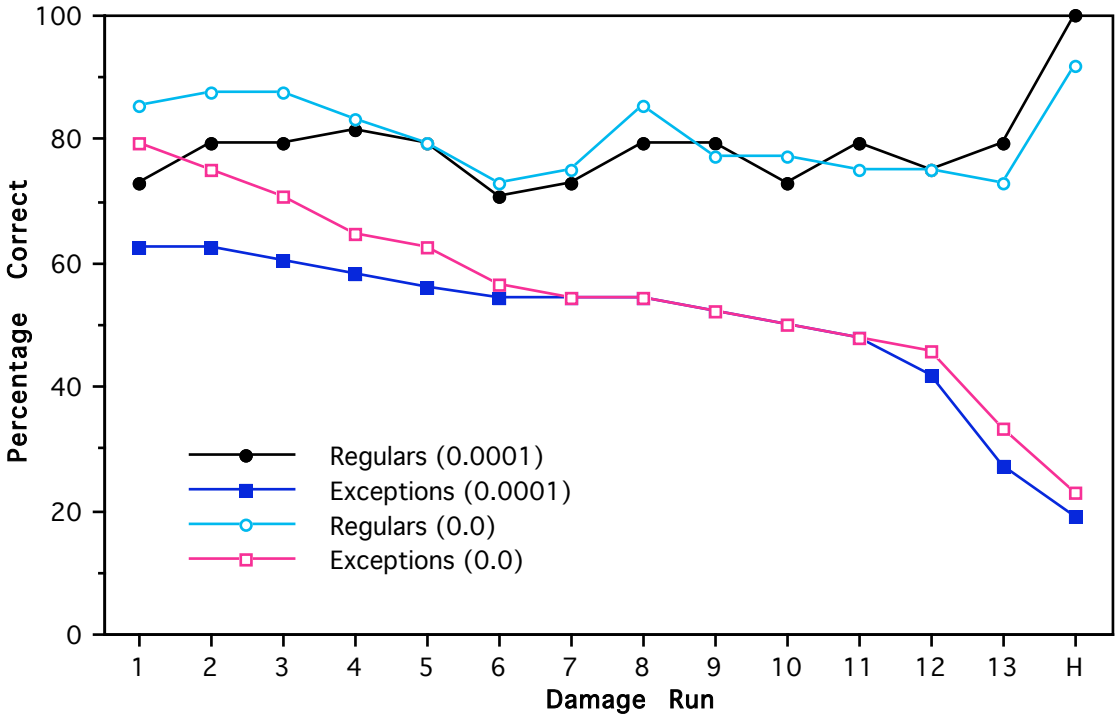

Figure 47 *An illustration of the range of variations in the size of dissociation for the removal of different sub-sets of hidden units. Plotted are the best dissociations found for each damage run of the errcrit = 0.0 and 0.0001 networks with the regular word performance better than 70%. For the final case (denoted H) the removed units were chosen by hand rather than at random.*